Computing today

DECEMBER 1983
85p

GETTING MORE FROM YOUR 64

Our new series provides fast sprite routines from BASIC

Learn to touch-type on the BBC Micro with our tutor program

Curve-fitting for mathematicians on the ZX Spectrum

Use machine code routines from BASIC easily with our Genie DATA statement generator

Our Dragon 32 "Graphicks" will simplify high-res drawing

FREE COMPETITION!
So you think you can write Adventures? Fame and fortune awaits you – see inside for details

BUILD YOUR OWN cortex

16 bit, 64K RAM colour computer

With this powerful machine (featured in Electronics Today International as a constructional project) you have access to highly advanced systems and software developed specially by MPE Ltd for the CORTEX. For business, education, R & D – or simply increasing your knowledge and understanding of computers – it beats comparably priced off-the-shelf machines hands down!

STATEMENTS IF ELSE ON GOTO GOSUB POP REM FOR NEXT ERROR INPUT PRINT ; ? * 1 UNIT BAUD CALL DATA READ RESTOR RETURN STOP TIME WAIT SAVE LOAD MOTOR ESCAPE NOESC RANDOM ENTER LIST PURGE NUMBER RENUM BOOT GRAPH TEXT PLOT UNPLOT COLOUR CHAR SPRITE SHAPE SPUT SGET MAG TOF TON DIM LET DEF NEW END BIT CRB CRF MEM MWD BASE

COMMANDS RUN SIZE CONT MON

DELIMITERS TO TAB STEP THEN ; @ # : , ? % $. ! & ( ) []

FUNCTIONS FNA-FNZ ABS ADR ASC ATN SIN COS EXP FRA INT LOG SQR SYS TIC SGN BIT CRB CRF MEM MWD LEN MCH POS COL MOD RND KEY

OPERATORS OR LOR AND LAND NOT LNOT LXOR == = > >= < <= <> - + / * ^

Self assembly kit

£295

Ready built £395

All prices exclusive of VAT. Carriage paid.

Optional extras

RS232C interface kit	£9.20
Floppy disc interface	£65.50
Pair of 5¼" disc drives	£300.00
Hardware kit & connectors	£49.50

Ready built

CORTEX B – Basic machine + RS232C	£410.00
CORTEX C – as above + disc drives	£895.00

Full assembly instructions and 216 page user's manual.

Standard features –

- High speed 24K byte extended basic interpreter
- Powerful TMS9995 16 bit microcprocessor
- 48 bit floating point gives 11 digit accuracy
- High resolution (256 x 192) colour graphics
- Screen memory does not use up user memory space
- 16 colours available on the screen together in graphic mode
- Fast line drawing and point plotting basic commands
- High speed colour shape manipulation from basic
- Full textual error messages
- String and Array size limited only by memory size
- Real time clock included in basic
- Interval timing with 10mS resolution via TIC function
- Named load and save of basic or machine code programs
- Auto-run available for any program
- Powerful machine code monitor
- Assembler and Disassembler included as standard
- Auto line numbering facility
- Full renumber command
- Simple but powerful line editor
- Buffered i/o allows you to continue executing the program while still printing
- Flexible CALL statement allows linkage to machine code routines with up to 12 parameters
- Basic programs may contain spaces between key words to make programs readable without using more memory
- Over 34K bytes available for basic programs
- Extended basic includes IF-THEN-ELSE
- Supports up to 16 output devices: Screen and cassette interfaces included as standard
- Supports bit manipulation of variables from basic
- Error trapping to a basic routine included
- Basic supports Hexadecimal numbers
- Separate 16K video RAM for graphics

POWERTRAN cybernetics

Portway Industrial Estate, Andover SP10 3CT. Tel: 0264 64455

To POWERTRAN CYBERNETICS, Portway Industrial Estate, Andover, Hants SP10 3NM.

Please send me ______________________________

____

I enclose cheque for ______ or charge to:

Access/ Barclaycard A/C No. [| | | | | | | | | | | | | | | | | | | |]

Name ______________________________

Address ______________________________

______________________________ Tel. ____________

Acting Editor:
Peter Green
Advertisement Manager
Malcolm Wynd
Advertisement Copy Control:
Sue Couchman,
Ann McDermott
Managing Editor:
Ron Harris BSc
Chief Executive:
T.J. Connell

Origination and design by MM Design & Print.

ABC Member of the Audit Bureau of Circulation

ISSN 0142-7210

Computing Today is normally published on the second Friday in the month preceding cover date. Distributed by: Argus Press Sales & Distribution Ltd, 12-18 Paul Street, London EC2A 4JS. 01-247 8233. Printed by: Alabaster Passmore & Sons Ltd, Maidstone, Kent.

Subscription Rates : UK £13.30 including postage. Airmail and other rates upon application to Computing Today Subscriptions Department, 513 London Rd, Thornton Heath, Surrey CR4 6AR.

Computing Today is constantly on the look-out for well written articles and programs. If you think that your efforts meet our standards, please feel free to submit your work to us for consideration.

All material should be typed. Any programs submitted must be listed (cassette tapes and discs will not be accepted) and should be accompanied by sufficient documentation to enable their implementation. Please enclose an SAE if you want your manuscript returned, all submissions will be acknowledged. Any published work will be paid for.

All work for consideration should be sent to the Editor at our Charing Cross Road address.

CONTENTS

VOL 5 NO 10 DECEMBER 1983

EDITORIAL & ADVERTISEMENT OFFICE
145 Charing Cross Road, London WC2H 0EE.
Telephone 01-437 1002-7. Telex 8811896.

CONSUMER NEWS....6
Find out what's new on the personal computer marketplace.

SOFT WARES........10
Interested in new software releases? Here's the page to look for the details.

BUSINESS NEWS.....14
Our monthly summary of the products available for the professional computer user.

MIKRO MAGIC......18
The MIKRO cartridge is much more than just an assembler for the Commodore 64. Our intrepid machine code programmers put it to the test.

DRAGON 'GRAPHICKS' 22
This piece of software will make life much easier for anyone wanting to use the high res graphics on the Dragon 32.

GETTING MORE FROM YOUR 64............29
The Commodore 64's BASIC is fairly — well — basic! Part 1 of this three-part series looks at sprites and how to make use of them easily.

MCBAS CONVERTER.38
The usual way of combining machine code with BASIC is to poke the code from DATA statements. This routine generates the DATA in the first place — error free.

ADVENTURES PT 2....45
Concluding our 'How to write them yourself' article, which looks at BASIC coding and will be useful for our . . .

COMPETITION 48
Now is the time for all good programmers to sit down at the keyboard and write an Adventure game. The winner gets an extra special prize.

LEARNING FORTH PART 2..............53
In this month's article we look at further stack manipulation commands and introduce the concept of variables, FORTH-style.

INDEX '83............61
Our superbly sorted guide to the past year's offerings from your favourite computer mag.

A BOLT FOR THE COMMODORE.......64
Tape operations on the Commodore 64 are usually a good cure for insomnia. The Arrow cartridge speeds things up and offers much more besides.

CURVE-FITTING 67
Mathematicians will familiar with the problem of drawing the best curve through a set of points. Here's a technique and a program that uses it.

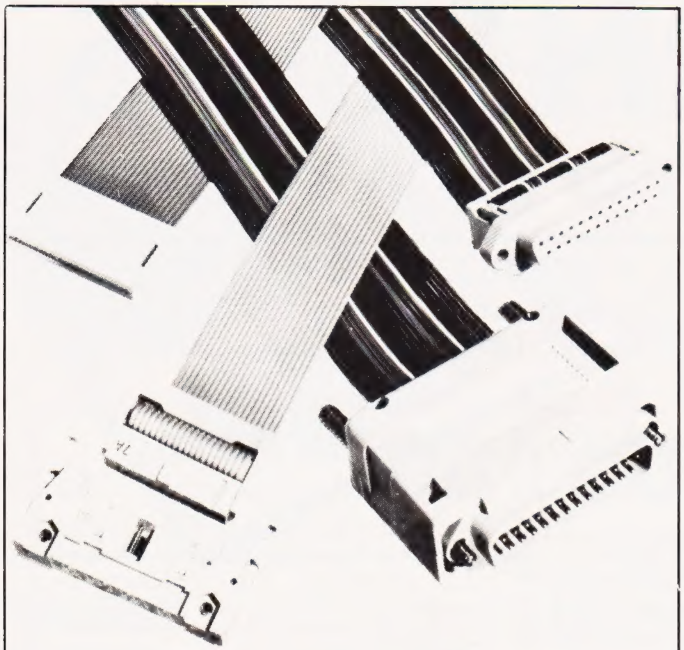
BBC TOUCH TYPING TUTOR..............71
The slowest digital operation on a computer is often that performed by the digits on the operator's hand! Train yourself to be faster and more accurate.

PROBLEM PAGE.....76
Our biggest program to date sorts things into sets, and a magic square is offered to puzzle you.

CT STANDARDS.....78

BOOK REVIEW.......80
This month our reviewer has his beady eye on books ranging from the Commodore 64 to the IBM PC and CP/M.

Next Month's Computing Today....17
ASP Software....................36
Sharp Microdealer...............79
Advertiser's Index..............83
Nascom Microdealer..............86
Classified Ads..................87
Computamart.....................89



Your Interconnection System for the Microcomputing World

Choose from our M50 range of exciting products all designed to assist the hobbyist in building an interconnection system most suitable for his particular application:

headers; sockets; colour coded cable; DIP connectors; sub-miniature D25 way plug, socket and hood.

We are exhibiting at the following exhibitions, so why not come along and see the products for yourself.

Our new catalogue containing over 150 new products is available from mid-October.

For further information on these products ring (04215) 62829 or write to:

BICC Vero Electronics Limited,
Retail Dept., Industrial Estate,
Chandlers Ford, Hampshire,
SO5 3ZR.

BICC vero

EVERYDAY DECEMBER ISSUE 90p

ELECTRONICS and computer PROJECTS

NOW ON SALE

TWIN CASSETTE INTERFACE FOR THE TRS-80

This add-on I/O Port enables the basic TRS-80 Model I Level II Microcomputer to be simultaneously linked with two cassette decks which until now was only possible with the expensive Expansion Interface. Both decks can be used for Saving, Loading and bi-directional file-handling operations.

ENVIRONMENTAL DATA RECORDER FOR THE ZX SPECTRUM

Stores data received from sensors in digital form in CMOS RAM for subsequent read-out via a computer. Particularly valuable where quantities being measured change slowly or readings are required over a long time period. Portable and weatherproof, ideal for field work. May be used with other computers with minor modification.

Happy Memories

Part type	1 off	25-99	100 up
4116 200ns	1.25	1.15	1.10
4164 200ns	4.20	3.95	3.85
2114 200ns Low power	1.15	1.00	.90
2114 450ns Low power	.95	.85	.80
6116/2016 150ns	3.35	3.00	2.85
6116 150ns Low power	4.90	4.40	4.20
2708 450ns	3.25	2.95	2.80
2716 450ns 5 volt	2.75	2.50	2.40
2716 450ns three rail	5.75	5.00	4.65
2732 450ns Intel type	3.45	3.05	2.95
2532 450ns Texas type	3.85	3.45	3.30
2764 250ns	4.20	3.95	3.75
27128 250ns	14.00	12.50	12.00

Z80A-CPU £2.99 Z80A-PIO £2.99 Z80A-CTC £2.99
6522 PIA £3.70 7805 reg .50 7812 reg .50

Low profile IC sockets:

Pins	8	14	16	18	20	22	24	28	40
Pence	12	13	14	16	18	22	24	27	38

Texas solder-tail:

Soft-sectored floppy discs per 10 in plastic library case:
5 inch SSSD £17.00 5 inch SSDD £19.25 5 inch DSDD £21.00
5 inch DSQD £26.35
8 inch SSSD £19.25 8 inch SSDD £23.65 8 inch DSDD £25.50

74LS series TTL, large stocks at low prices with DIY discounts starting at a mix of just 25 pieces. Write or 'phone for list. Please add 50p post & packing to orders under £15 and VAT to total. Access & Visa welcome. 24hr service on (054 422) 618. Government & Educational orders welcome, £15 minimum. Trade accounts operated, 'phone or write for details.

Happy Memories (CT), Gladestry, Kington, Herefordshire. HR5 3NY. Tel: (054 422) 618 or 628

AT LAST, A HOME COMPUTER THAT IMPROVES WITH AGE.

It's surprising how many first-time relationships with a home computer go sour with age.

You buy an attractive, discounted little machine so that you and the children can learn about computers.

Instead, you learn about its limitations: the dull graphics. The plugs that fall out. The cheap power supply. The unalterable "beginners" language. The stiff, fragile keys. No provision for future developments. If only you'd looked around a bit in the beginning... *"Quality costs a little more, but it's usually worth paying for"* (Personal Computer News – CGL M5 Review, June '83.)

The CGL M5 is designed and built by Sord, one of Japan's leading computer specialists, with three main ideas in mind.

First, to be easy and fun to learn and operate.

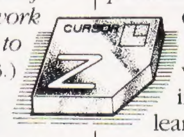
Second, to be rugged enough to last through hours and hours of operation.

And third, to form the basis of a powerful, versatile home computer system that won't need replacing until you're ready for a dedicated business system.

Built to learn

The CGL M5 is designed to be easy for non-geniuses to use.

"On the M5, most of the work is done for you, and all that is left is the need to work out what to do next, rather than how to do it." (Personal Computer News, June '83.)



If you make a mistake, you can correct it with a simple movement of the cursor. So you only correct that mistake, not a whole line; nor do you have to indulge in complex edit commands.

Budding video game designers and computer artists will love to get their hands on the 16 colour graphics and 32 moveable images called "sprites."

"The M5 makes professional graphic effects very simple for even the beginner to achieve." (Personal Computer World, Aug. '83.)

Built to last

"It works first time, doesn't need a lot of mollycoddling and jiggery-pokery to persuade it to continue to do so, and what's even better, it continues to work well. You don't have to balance cold cartons of milk on the top, shove matches in the back to keep the plugs in, or press the keys with several pounds force to make them respond." (Personal Computer News, June '83.)

Being able to build things that work and carry on working without endless maintenance is something at which the Japanese seem to excel.

Built to grow

To be truly versatile, a home computer has to understand very different things.

So you need different "languages," which the M5 provides by supplying part of its memory in plug-in cartridges.

"The M5 eliminates the worst limitations on machines at this level, which is that they tend to be stuck with whatever language is provided by the management." (Personal Computer News, June '83.)

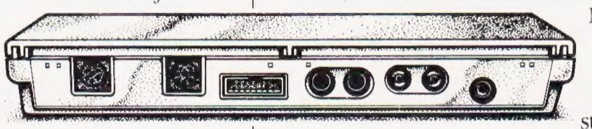
The computer is supplied complete with a Basic-I cartridge, a standard integer BASIC language and a simple learning text.

Plug in the Basic-G cartridge, and you can access the M5's incredibly sophisticated graphic and sound capabilities which are far in advance of similarly-priced computers.

Move on to the Basic-F cartridge, and you have scientific, technological and statistical computing power usually available only on big computers with equally big price tags.

The FALC cartridge provides a tailor-made language for data management, spreadsheet accounts and business problems. Combine FALC with a disc and you could *"turn the M5 into a small business machine"*. (Personal Computer Magazine, August '83.)

Now, take a look at the back of the M5.



Notice the sockets (usually an extra) for a standard Centronics-type printer, the separate video monitor and hi-fi sound output.

Even the language cartridge socket has hidden potential:

"Unlike most such sockets, this one has 56 internal lines connected to it giving access to just about every function in the computer. This means that just about everything you can think of can be added onto the computer, ranging from a Prestel interface to second processor to use as an intelligent terminal on a timesharing computer"... (Electronics – The Maplin Magazine, March '83.)

Take a look at the home computer that will improve with age.

For a full technical specification of the CGL M5, details of the wide range of supporting software and to find out where to see a complete demonstration, send the coupon to:
CGL, CGL House, Goldings Hill, Loughton, Essex IG10 2RR. Telephone number: 01-508 5600.

I'd like to know more about the CGL M5. Please send me a brochure and a list of dealers. CT2

Name________________

Address________________

CGL M5 HOME COMPUTER

BUILT TO LEARN. BUILT TO LAST. BUILT TO GROW.

CONSUMER NEWS

WALKIES!

Anyone who wants to take their BBC Micro for a stroll can now do so in style. Intastor Micro Aids of Stroud, Gloucester have introduced a custom-built carrying case to their official range of BBC Microcomputer support products: manufactured in tough, durable fibre board with smart black finish, the new case is designed to accommodate the computer and all its accessories. The inner fitments are vacuum-formed in an engrained ABS finish to give separate compartments for computer, all connecting leads, a cassette player (to maximum measurements of 11" x 7" x 3"), software cassettes and the course handbook. The case has a removable lid, interior foam protection, protected corners, plated locks and comfortable carrying handle. It measures just 28½" x 22" x 5" and is surprisingly lightweight when fully packed.

This is the latest addition to a growing list of Intastor BBC Microcomputer support products which also include the Official BBC Programmers' Kit, two sizes of print-out binders and a programmers' grip binder. For full details of prices, or if difficulty is experienced in location stockists, please apply direct to Intastor Micro Aids, FREEPOST, Stroud, Glouchester GL6 0BT (telephone 045 383 2334).

PRETTY FOXY

The new Zedxtra intelligent joystick interface for the Sinclair ZX Spectrum, introduced at the last ZX Microfair, will now be distributed under their own name by Fox Electronics Ltd, of 141, Abbey Road, Basingstoke, Hants RG21 9ED (telephone 0256 20671), and all enquiries should be sent to them.

The interface allows all software designed for use with the keyboard to be used with a standard 9-pin joystick, such as Atari. It consists of a small box which plugs into the rear socket; an extension socket is provided for further expansion. A switch on the box turns on the interface; this results in a hardware reset to a program held in a CMOS RAM, the contents of which are maintained by a trickle-charged battery. The program puts a menu-selection display on the screen, and the user can use the keyboard to enter the names of up to 16 games, and the keys to be used for the five main joystick functions, up, down, left, right, and fire. After the user has selected the game to be played, the program calculates values for all eight directions, with and without the fire button, loads all the data into the CMOS RAM, and resets to BASIC.

The user then loads the games tape in the normal way. The keyboard is usable unless the joystick is actually being operated, when the correct codes are sent to the Spectrum from the CMOS RAM. When a new game is desired, the interface switch is turned off; this generates a hardware reset, meaning that there is no need to power-off between games. With the switch off, the Spectrum acts as if the interface is not fitted. Turning the switch back on returns to the menu program. All the data in the RAM is saved with power off. To provide security, a back-up tape of the RAM contents can easily be made, and this allows unlimited sets of up to 16 games each to be kept on tape.

The interface can also be used as a pseudo-ROM; machine code routines can be loaded into the CMOS RAM, and these can be made available above RAMTOP when BASIC is running. This could be used to store a programmer's toolkit, or user-designed characters, for example. The unit is compatible with Microdrives and printer, and has its own power regulator to avoid overloading the Spectrum regulator. The price is £28.50 all inclusive.

ATARI'S XL-5

Atari International (UK) Inc has announced the UK prices and availability dates for the first in the range of Atari XL home computer products. The Atari 6000XL home computer with 16K RAM (expandable to 64K) should be available in the UK by the time you read this with a recommended retail price of £159.99. Its technical specifications include full-stroke design keyboard with international character set, built-in Atari BASIC, Help key, software compatibility with all Atari software, 11 graphics modes and four independent sound voices.

This will be complemented by five peripheral devices. The Atari 1020 Colour Printer, which is perfect for creating charts, graphs and artistic designs, has a recommended retail price of £199.99. The Atari 1025 80-Column Printer is dot-matrix printer which offers a choice of print styles, from a standard 80 characters per line, to condensed type at 132 characters per line, to an extra-bold 40 characters per line. For use with typing or computer paper this printer will be available in November at a recommended retail price of £349.99. The Atari 1027 Letter Quality Printer prints fully-formed letters like a quality electric typewriter. Ideal for use with a word processor like the AtariWriter, this printer will be available in October at a recommended retail price of £49.99. Finally, the Atari 1050 Disk Drive offers a dual-density format for fast access to information and has a recommended retail price of £299.99.

COMMODORE PRICE CUT

Commodore have asked us to point out that the price of the Commodore 64, given in their advertisement in the October issue as £299, should have read £229.

LOOKS GOOD ▲

Apple have introduced a new full-feature green screen monitor for the Apple IIe personal computer system. The new monitor incorporates a number of custom-designed features to benefit Apple IIe users. Monitor II is designed to aesthetically with the Apple IIe, and has a tilt screen facility which allows it to be adjusted to be perpendicular to the operator. This reduces annoying reflections from overhead lightning. Further reduction of reflections from ambient light sources is possible due to the monitor's anti-reflective screen surface.

The new monitor has been designed for use with all video display applications. These include spreadsheet, word processing, and all 40 or 80 column monochrome applications. Monitor II connects to the Apple IIe via a standard video cable which is supplied with the computer. If you don't know your nearest Apple dealer, try contacting Apple Computer (UK) Ltd, Eastman Way, Hemel Hampstead, Herts HP2 7HQ (telephone 0442 60244).

BBC-ZX PRINTER INTERFACE

Now BBC owners can get into print for less than £70. This new interface allows the ZX Printer to be used with the BBC Micro, making it the cheapest printer option on the market today and, therefore, ideally suited to both the younger user and schools (Spectrum and BBC can now share the same printer).

The interface comes complete with relocatable machine code software on cassette together with full instructions and requires no modifications to either BBC Micro (connected via the 1 MHz Bus) or the ZX Printer.

The Interface contains its own mains derived power supply which provides 9 V and a regulated 5 V for the ZX Printer and internal decoding network. The unit is available for £29.95 including VAT and postage direct from W.D. Interfaces, 12 Leabank Avenue, Garforth, Leeds, LS25 2BL (telephone Leeds 864328) or from selected outlets.

MZ-700 CHARACTER CORRECTION

We got it wrong! The information regarding the alternative character set in the Sharp MZ-700 review in the October issue was totally incorrect. We won't embarrass the gentleman who gave us the wrong details by revealing his identity, but the method of accessing the other 256 characters is not by using POKE, but by setting the top bit (bit 7) in the screen colour memory corresponding to the screen location you wish to change. Colour memory is from \$D800 to \$DFFF. The program below will put all the normal characters onto the screen, then go through changing them one at a time to the alternative. (Trying this on the review machine still didn't give us the correct characters: we got Russian and Greek alphabets and some garbage shapes. Production models have 'skeleton' letters and a range of spaceships, chess pieces and similar graphics.)

```
10 CLS: A=0
20 FOR I=0 TO 255
30 A=A+2: IF INT(A/40)=A/40 THEN A=A+40
40 POKE $D000+A,I
50 NEXT I
60 A=0
70 FOR I=0 TO 255
80 A=A+2: IF INT(A/40)=A/40 THEN A=A+40
90 POKE $D800+A,(PEEK($D800+A) OR 128)
100 FOR J=0 TO 500: NEXT J
110 NEXT I
120 GOTO 120
```

AT LONG LAST DISCS

Dragon Data Ltd has, at last, launched a disc drive unit for the Dragon 32 home computer. Priced at £275, the Dragon Disc Drive is a single half-height drive in a coated steel case. It has an internal power supply and is easily expandable to a double disc system by inserting an additional drive. Two double units can be linked to form a four-drive system.

Its specifications are as follows. The system uses 5¼" diskettes and the memory capacity (formatted) is 184,320 bytes. The disc organisation is single-sided, double-density, with 40 tracks, 18 sectors per track, and 256 bytes per sector. The directory is on track 20. The controller can support up to four drives, single or double-sided capability. Up to 10 files may be open simultaneously.

The Dragon Disc drives will be available through the usual Dragon dealerships and retailers, including Boots and Dixons.

BRIEFING

A purpose-designed 14 inch colour monitor at under £200 heralds the entry of Fidelity, the West London consumer elctronics company, into the computer industry. The **CM14 monitor** accepts either RGB, RGBY or composite video inputs, together with audio: thus the CM14 will interface with virtually any computer or games machine capable of driving a monitor, or work equally well with VCR, disc, cable data or satellite adaptors. At under £200, the price compares very favourably with less sophisticated monitors costing nearly double. This has been achieved by using the latest state-of-art IC technology, with minimum component count, together with the company's high volume manufacturing capability. More details from **Fidelity plc,** Victoria Road, London NW10 6ND (telephone 01-965 8771).

Improved production techniques employed in Epson, Japan, coupled with increased demand for its range of dot matrix printers, has enabled **Epson (UK) Ltd** to reduce the price of its established printers and to introduce new products, notably the company's latest **FX-100** and **RX-80 FT**, at a lower price than would be possible otherwise. A reduction of as much as 10 per cent has been achieved in some cases and the recommended retail prices are now as follow: MX-100 £475.00; RX-80 T £279.00; FX-80 £438.00; RX-80 FT £319.00; FX-100 £569.00. All the above prices are exclusive of VAT.

WATCH OUT for the **VIRGIN GAMES FUN BUS** – on tour **NOW**

Angler
by Dirk Olivier
Spectrum 48K
VGC 1012
Fishy fun for all ages – but don't catch a crab!

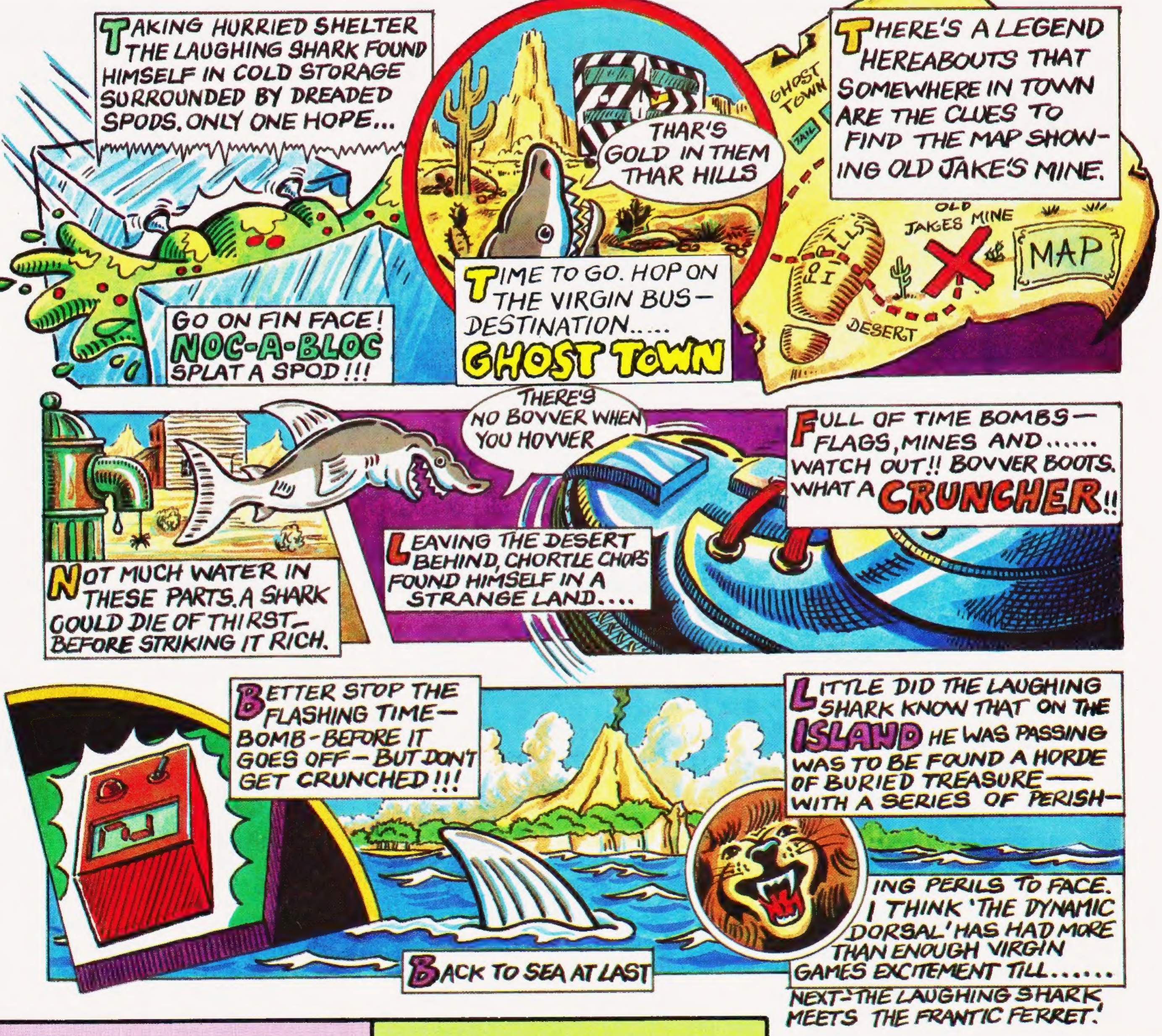
Rider
by Roy Poole & Terry Murray
Spectrum 48K
VGC 1014
Parachute into the enemy territory and ride the mined roads.

Plankwalk
by Neil Cannon
BBC B
VGA 2008
Can you help scaffolding Sid to stay alive!

UXB
by Patrick Fisher
Dragon 32
VGB 4004
A nail-biting test of skill and nerves to defuse the unexploded bomb.

Noc-A-Bloc
by Richard Bygrave
BBC B
VGA 2010
Just when you thought it was safe to go back in the deep freeze!
Joystick/Key

Ghost Town
by John Pickford
Spectrum 48K
VGC 1013
An intriguing graphical adventure

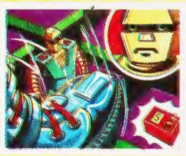


MAIL ORDER

All our programs are available at normal retail price including postage and packing direct from our **"MY LOCAL DEALER STILL DOESN'T STOCK YOUR PROGRAMS DESPITE THE FACT THAT THEY'RE REALLY GOOD DEPARTMENT"** at 61-63 Portobello Road, London W11.

COLOUR CATALOGUE

If you want a copy of our sixteen page colour catalogue **FREE** listing details of all our games, please write to the **"GIMMEE A CATALOGUE QUICK JIMMEE DEPARTMENT"** enclosing an A4 SAE (with 17p stamp).



Cruncher
by Malcolm Ripley
BBC B
VGA 2009
Trample the time-bombs, but avoid the boots or be 'crunched'.
Joystick/Key

The Island
by Martyn Davies
Spectrum 48K
VGC 1015
Find the treasure – a full 48K adventure with action sequences.

THE "I WANT TO BE RICH AND FAMOUS DEPARTMENT"

We are always keen to receive any original entertaining programs with good graphics at our now famous **"I WANT TO BE RICH AND FAMOUS DEPARTMENT"** from any of you programming mega-stars out there. Don't delay – send today!

VIRGIN GAMES GANG

Our **GANG** is growing in numbers all the time, and everybody who buys one of our **NEW GAMES** will **RECEIVE:**

- a year's **FREE MEMBERSHIP** of the **GAMES GANG;**
- **FREE ENTRY** in the next **VIRGIN GAMES GANG DRAW;**
- **6 FIRST PRIZES,** consisting of a **VIRGIN DAY OUT** – a trip on the **VIRGIN GAMES FUN BUS** to the **VIRGIN MANOR RECORDING STUDIO** in Oxfordshire. And then be our guest at the famous **KENSINGTON ROOFTOP NIGHTCLUB, "THE GARDENS";**
- **50 SECOND PRIZES** of **VIRGIN GAMES POSTERS.**
- **100 THIRD PRIZES** of **VIRGIN GAMES POSTERS.**

VIRGIN GAMES ARE:

Vic 20	**MISSION MERCURY**	£7.95
Vic 20	**ENVAHI 8K**	£5.95
Vic 20	**CREEPERS 3/8K**	£5.95
Spectrum	**YOMP 16/48K**	£7.95
Spectrum	**STARFIRE 48K**	£7.95
Spectrum	**SHEEPWALK 48K**	£7.95
Spectrum	**GOLF 16/48K**	£7.95
Spectrum	**LOJIX 48K**	£5.95
Spectrum	**RACING MANAGER 48K**	£5.95
Spectrum	**QUETZALCOATL 48K**	£5.95
Spectrum	**ROBBER 48K**	£5.95
Spectrum	**SPECTRON 48K**	£5.95
Spectrum	**LOST 48K**	£5.95
Spectrum	**THE ISLAND 48K**	£5.95
Spectrum	**GHOST TOWN 48K**	£5.95
Spectrum	**RIDER 16/48K**	£5.95
Oric	**KILLER CAVERNS 16/48K**	£6.95
Dragon	**DEATH CRUISE**	£6.95
Dragon	**I CHING**	£6.95
Dragon	**CASTLE ADVENTURE**	£6.95
Dragon	**UXB**	£6.95
COMMODORE 64	**FALCON PATROL**	£6.95
COMMODORE 64	**BITMANIA**	£6.95
BBC B	**BUG BOMB**	£7.95
BBC B	**LANDFALL**	£7.95
BBC B	**SPACE ADVENTURE**	£7.95
BBC B	**TRENCH**	£7.95
BBC B	**'OWZAT**	£7.95
BBC B	**CHIEFTAIN**	£7.95
BBC B	**MICROBE**	£7.95
BBC B	**PLANKWALK**	£7.95
BBC B	**CRUNCHER**	£7.95
BBC B	**NOC-A-BLOCK**	£7.95
TI99/4A	**ROBOPODS**	£6.95
TI99/4A	**FUN-PAC**	£6.95

SOFT WARES

GAMES GALORE

Spectacular 3D, an entirely new generation in computer games and sophisticated graphics and some of the features in Quicksilva's new range of 11 computer games which are aimed to hit the Christmas best-sellers.

Games Designer — the first product to come out of Quicksilva's Software Studios, is an entirely new idea. Users of the 48K Spectrum can program almost any type of arcade game and no programming experience is necessary. They can begin by using the eight programs, such as the Mutant Hamburger or Turbo Spider. (Look, we don't write this stuff, we only report on it! Ed.) The player can then alter individual aspects such as the appearance of a character, then move on to changing the background, altering the speed of attacks or changing sound.

They can then progress to writing original games and the Games Designer also has a more sophisticated use — as a development tool to check out the visual appeal of a design. Priced at £14.95, a useful manual is included in the package. The author is John Hollis, who designed the successful game Timegate. We've already played with the Games Designer and it's pretty impressive.

3D is featured in Quicksilva's new Soft Solid 3D Ant Attack. The game enables the player to walk around the Walled City Antescher: this is displayed in solid 3D blocks which can be viewed from any angle. Ant Attack is for the Spectrum 48K and will cost £6.95.

Quicksilva expect Purple Turtles to be a best-seller at Christmas (but will it replace turkeys?). Described as "the cutest game around" (oh dear), it is expected to appeal to all ages. Designed for the Commodore 64, it will retail at £7.95.

The sophisticated Generators for the BBC Model B enables the user to make up high quality multi-coloured characters to be used in any program. It can also be used as a teletext generator — it allows the user to build up pictures in the teletext mode and has the added advantage of using very little memory. Written by Dave Mendes, author of the successful Beeb Art, it will cost £6.95. The famous Mined Out is now also available for the BBC — Bill the Worm rules OK.

All Quicksilva's games are available by mail order or from major retail outlets and the company is guaranteeing retailers a fast efficient delivery service. Quicksilva are at 13 Palmerston Road, Southampton, Hampshire SO1 1LL (telephone 0703 20169).

EPSON INFORMATION

Epson (UK) Ltd has published two new software catalogues, one for the HX-20 portable computer and one for the QX-10 desk top microcomputer. The catalogues have been compiled as an aid to Epson users and list both the software distributed directly by Epson, along with that developed by other suppliers for use with Epson equipment. The catalogues list the packages under their application headings and include prices and names of dealers from whom they are available as well as a brief description on how each one works.

Epson would welcome details of new packages from software authors who would like to be considered for inclusion in the next updated issue. Authors should write to the Editor, Software Catalogue, Epson (UK) Ltd, Dorland House, 388 High Road, Wembley HA9 5UH. Software catalogues are available direct from Epson (UK) Ltd.

AUDIOGENIC ATTACK

Audiogenic Ltd — worldwide distributors of computer games — have announced their intention to capture a substantial share of the Texas cartridge games market. Texas Instruments have maintained a monopoly on the market for cartridge games for their TI-99 home computer, but they will soon be facing strong competition. Audiogenic will be expanding their field of operations and are predicting to take a substantial share of the Texas cartridge games market within the next 12 months.

Audiogenic's initial launch of new Texas games for the TI-99 will include 'St Nick', 'Cave Creatures', 'Rabbit Trail', 'Driving Demon', and 'Hen House', and the range will gradually be extended — details and prices to be announced shortly. It remains to be seen what will happen when Texas re-design the TI-99/4A so that it will only work with cartridges containing their (licenced) graphics ROM, as they have been threatening to do.

IT'S THE BYTLES

Husband and wife team Alan and Soo Maton aim to bring a new 1980s meaning to the 'Merseybeat' sound of the '60s — this time with the zap, pow and zing of computer games. They have formed a new software publishing house, Acme Software Ltd, as part of a boom in home computing which is giving the city much the same 'hotbed' image for computer gaming as it had for pop music with the Beatles. (The trouble is, whenever I read 'Acme' I think of Chuck Jones' Roadrunner cartoons. Oh, well...Ed.)

Existing Liverpool-based companies like Liversoft, Bug-Byte, and Imagine already account for a lion's share of the UK's estimate annual market of two-million computer games cassettes. Now Acme is bidding to add its own array of new games titles for all the main home computers. The founders have come to Acme via Bug-Byte. They see their new venture as "more of a friendly departure than a break-away" and intend Acme to be very much a software publishing house rather than an intense games programming base.

First three games to be launched under the Acme label include a good implementation of that familiar classic Connect Four — included because it is one of the first for the so-far thinly supported

48 or 96K Lynx. Then there is Practically Impossible for the 16 or 48K Spectrum, in which a peculiarly blob-like Cyclops Eye has to be manoeuvred through gaps in five progressively complex screens of barriers. Finally a novel new game for the evergreen VIC 20, Bridgeman, involves an increasingly frantic attempt to plug the gap in six bridges as the construction firm boss makes an irate (and utterly random) inspection of progress.

Acme promise that all their games will cost £5 each (including postage and packing during the early stages of mail-order-only supply). High Street distribution through a dealer network will follow, as will some tapes (still at £5) with more than one game on each.

For the future, Acme already has three more games on the stocks, including an adventure for the Commodore 64 called Altair 4, occupying 52K and written in machine code for faster reactions; an "exciting novel" new game for the unexpanded VIC, called Mega-Vault; and Mowermania, this time for the VIC with 8K expansion. Acme Software Ltd are at 49/51 The Albany, Old Hall Street, Liverpool, Merseyside, L3 9EJ (telephone 051-236 8062).

APPLE ACCOUNTING

Nibbles Systems of Bournemouth have introduced Easy Banker in recognition of the growing demand, from dealers and micro users alike, for an inexpensive yet comprehensive accounting package. Easy Banker is the first Nibbles-produced package for the Apple which will be launched to satisfy the needs of users who wish to gain experience in computerised systems before fully defining their final requirements. The low cost of the package also allows dealers to increase their sales performance by offering comprehensive systems at a price that was previously impossible. It is particularly attractive to the first time user as a comprehensively analysed cash book system, or the larger business department as an effective tool for solving the ever present problem of petty-cash accounting.

Nibbles hope to strengthen their presence in the marketplace over the next few months and are keen to hear from interested dealers and users alike. For further information contact: Nibbles Systems Ltd, 49 Poole Road, Westbourne, BOURNEMOUTH BH4 9BA (telephone Bournemouth 768600).

WORDS ON WORDCRAFT

Wordcraft Designs Ltd has now released its 16-bit version of the highly successful Wordcraft word-processor. The first three machines supported are the ACT Sirius 1, the IBM-PC and the IBM-PC-XT.

Wordcraft, through its distributor Dataview of Colchester, has sold over 15,000 copies on the Commodore range of micros and earlier this year it was awarded the ICP multi-million dollar award for software sales. Wordcraft Designs has taken the opportunity to redesign the software so that it can be easily configured for a wide range of new 16-bit machines. To this end over 90% of the system is now written in BCPL, a systems programming language developed at the University of Cambridge, while the machine-dependent and high-speed parts of the system are coded in 8086 assembler.

Wordcraft Designs say they are ahead of competing products in a number of ways. First, they use all the features of the machines they support; for example on the Sirius they not only have multiple screen menus for all the function keys, but make full use of the software-configurable keyboard and character set. On the IBM-PC they use of the function keys and allow access to the full character set.

Many features either not found on other products or available only as extra programs are included as standard. For example, spelling checking, mailing-list management and selection, arithmetic, and background printing are all standard. The fully interactive spelling checker is unique, claim Wordcraft — spelling can be checked immediately from anywhere in a document, any misspelt word being indicated in the text while the section of the dictionary that should contain it is visible in a screen window. The correct word may be automatically taken from the dictionary or new words added to it. This removes the inconvenience of running spelling checkers as batch jobs by making a check available at any time.

Wordcraft supports all possible printers for output. The Printer Definition Files mean that Wordcraft can make full use of the advanced facilities of today's printers capable of incremental horizontal movement.

The unique configuration file means that versions of the software can be quickly tailored for overseas markets with all screen messages and keyboard layouts easily translatable. No reprogramming is required for different language versions. Help is always available either as special 'help' screens, function key menus or screen prompts making the system friendly to the new user.

It is Wordcraft Designs' intention to make Wordcraft as much a standard as the popular Wordstar package. "We are looking for deals with manufacturers of micros so that Wordcraft can be supplied as a standard with new machines. The Osborne has set the trend by offering the complete business solution, hardware and software, in one package and we feel that we can offer manufacturers a superior product in the word-processing field" said Mike Lake of Wordcraft.

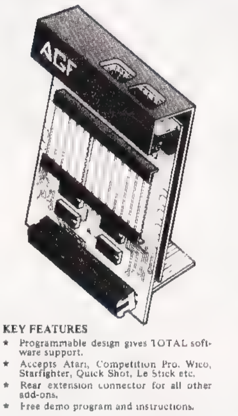
Wordcraft is supplied to Sirius and IBM dealers through Dataview Ltd of Colchester. A hotline service is available to assist dealers in answering any queries relating to the product. For more information please contact: Wordcraft Designs Ltd, 43 Farley Road, Derby DE3 6BW (phone 0332 683892), or Dataview Ltd on 0206 869414.

AGF

PROGRAMMABLE JOYSTICK INTERFACE for Spectrum or ZX81

ONLY 32·95 +£1·00pP

MICRODRIVE COMPATIBLE



ABOUT OUR INTERFACE

The AGF Programmable Joystick Interface is a unique design offering the use of any Atari-compatible joystick with absolutely all software, whether it is cassette or ROM cartridge, with the Sinclair Spectrum or ZX81.

The hardware programmable interface requires no additional software and accurately replicates the keys of the computer in a manner which is responsive to absolutely *ALL* key reading methods, both BASIC and Machine Code.

The interface does not interfere with key operation and can therefore be used simultaneously with the keyboard.

There is no need to remove the interface once fitted as the rear extension connector will accommodate further expansion, i.e printers or RAM packs etc. This important feature avoids excessive wear to the expansion port.

The key replication principle pioneered by AGF means that your own programs can use eight directional joystick movement by utilising simple key reading BASIC.

Two joystick sockets are provided which share the same keys, for use with the majority of two player games. Several interfaces may be used on the same computer for multiple joystick applications.

The interface is programmed by a two digit code, which is looked up on a programming chart supplied, for each direction and firing button. The two numbers are then selected on a pair of leads which are clipped onto appropriately numbered strips on the interface.

Once configured this can be marked on a Quick Reference Programming Card for storing with the game. As the programming is *not* power dependent the interface retains the last configuration made and can be immediately used when next switched on.

KEY FEATURES

- ★ Programmable design gives TOTAL software support.
- ★ Accepts Atari, Competition Pro, Wico, Starfighter, Quick Shot, Le Stick etc.
- ★ Rear extension connector for all other add-ons.
- ★ Free demo program and instructions.

PACKAGE CONTENTS SUPPLIED

- Programmable Interface Module as illustrated, complete with clip-on programming leads.
- Self adhesive programming chart detailing how to define which key is simulated by UP, DOWN, LEFT, RIGHT, and FIRE. This can be fixed on to the case of your computer or if preferred the protective backing can be left on. The chart is made of a very durable reverse printed plastic and is extremely easy to read.
- One pack of ten Quick Reference Programming Cards for at-a-glance setting to your games requirements. The card allows you to mark the configuration in an easy to read fashion with space to record the software title and company name.
- Video Graffiti demonstration program which is written totally in BASIC to illustrate how all eight directions and fire can be read. This is also a useful high resolution drawing program.
- 12 months guarantee and full written instructions.

JOYSTICKS

CONTROLLERS

FOR USE WITH OUR INTERFACE Module or VIC 20, Commodore 64, Atari VCS, Atari 400, Atari 800

If you require extra Joysticks for our original interface module mark order 'OLD' Joysticks

ONLY £7.54 inc VAT + P&P

FROM MR/MRS/MISS

ADDRESS

SEND C.W.O. (NO STAMP NEEDED) TO: A.G.F HARDWARE, DEPT.**CT**,

FREEPOST, BOGNOR REGIS, WEST SUSSEX, PO22 9BR

QTY	ITEM	ITEM PRICE	TOTAL
	PROGRAMMABLE INTERFACE	33.95	
	JOYSTICK(S)	7.54	
	PACK(S) QUICK REFERENCE CARDS	1.00	
ONE	VIDEO GRAFFITI	FREE	
ZX81 ☐	ZX SPECTRUM ☐ Please tick	FINAL TOTAL	

DEALER ENQUIRIES WELCOME *EXPORT PRICES ON APPLICATION*

AB & C Computers

MAIL ORDER SPECIALISTS

SEND LARGE SAE FOR LISTS INSTANT CREDIT FOR CALLERS

A SELECTION FROM OUR EXTENSIVE STOCK

★TEXAS SPECIAL OFFER PACKAGE★

T199 4A Computer + Extended Basic including Delivery **£165.95** whilst stocks last

Titch Software for T199/4A Hangman/Victory 4 Super Value at **£8.00**

Protek Spectrum Joystick Interface **£14.95** inc P&P Uses most switch type Joysticks

New Oric Joystick Interface **£14.95** inc. P&P

New MCP-40 Centronic 4 Coloured Printer fits most personal Micros **£149.95** inc P&P

ZX Spectrum & BBC Upgrades

Now in stock DRAGON DISC DRIVES £275.00 + £5. Delivery

New low prices on Texas T199/4A Contact us first

Are you short of a cassette recorder for Xmas — for your computer CONTACT US

Send Cheque with order made payable to AB & C Computers

Duchy House, 6 Lower Aylmer Square, St Austell, Cornwall.

ANNOUNCING

THE QUILL

FOR THE 48K SPECTRUM AT £14.95

The Quill is a major new utility written in machine code which allows even the novice programmer to produce high speed machine code adventures of superior quality to many available at the moment without any knowledge of machine code whatsoever.

Using a menu selection system you may create well over 200 locations, describe them and connect routes between them. You may then fill them with objects and problems of your choice. Having tested your adventure you may alter and experiment with any section with the greatest of ease. A part formed adventure may be saved to tape for later completion. When you have done so the Quill will allow you to produce a copy of your adventure which will run independently of the main Quill editor, so that you may give copies away to your friends. The Quill is provided with a detailed tutorial manual which covers every aspect of its use in writing adventures.

It is impossible to describe all the features of this amazing program in such a small space, so we hve produced a demonstration cassette which gives further information and an example of its use. This cassette is available at £2.00 and the Quill itself is £14.95.

ALSO NEW FOR THE 48K SPECTRUM:

DIAMOND TRAIL £4.95

The latest of our machine code adventures sets you the task of recovering the Sinclive diamond. But first you must overcome many problems in a city fraught with danger and intrigue.

GILSOFT

30 Hawthorn Road, Barry
South Glam CF6 8LE
Tel: (0446) 736369

TELEPHONE YOUR ORDER WITH

Access VISA

OUR SOFTWARE IS AVAILABLE FROM MANY COMPUTER SHOPS NATIONWIDE, OR DIRECT FROM US BY POST OR PHONE. S.A.E. FOR DETAILS. DEALER ENQUIRIES WELCOME. SOME OF OUR MAIN WHOLESALERS ARE:

UK:	PCS Distribution, Darwen, Lancs. Tel (0254) 691211/2
HOLLAND/BELGIUM:	AASHIMA TRADING BV, Hoogstraat 69a, 3011 PH Rotterdam
SWEDEN	RIKO DATA, Box 2082, S-230 41 Bara, Sweden
DENMARK.	QUALI-SOFT, Vesterbrogade 127 E Mz Tv, 1620 Copenhagen V
SOUTH AFRICA:	UNIVERSAL SOURCES (PTY) LIMITED, Durban, Natal

Map 80 Systems Ltd

NEW

MPI Multi Purpose Interface
Will control a mixture of 8″/5″/3″ drives
On board SASI/Winchester interface
Programable RS232 interface
Programable RS422/PN1360 high speed multi drop serial interface
On board CTC and SIO

Price	Kit	B&T
Complete MPI	£150	£185
FDC & SASI only	£125	£145

VFC Video Floppy Controller
80 col by 25 line display
On board floppy disk controller
Keyboard port
Video switch

Price	Kit	B&T
VFC Complete	£175	£214
Video only	£89	£110
FDC only	£95	£115

MAP RAM 256K RAM card
64/32K pages or full memory mapping

Price	Kit	B&T
64K version	£105	£150
256K version	£225	£285

AVAILABLE SOON

CPU MAP 80 CPU card
Memory mapped
64K RAM
2 fully programable serial channels (RS232)
Keyboard port
16 parallel I/O lines
CTC & SIO. Full interrupt control

Half height TEAC disk drive systems available from **£299**
CP/M 2.2 operating system **£110**. CP/M 3 now available **£257**
All prices exclude P&P (£1.50) & VAT
Dealer & OEM enquiries welcome
M.A.P. 80 Systems Ltd., No. 1 Windsor St., Chertsey, Surrey
Tel: 09328 64663

BUSINESS NEWS

A FLOPPY DRIVE?

Sternstat Ltd is marketing a new strong floppy disc case which is designed to contain thirty 5¼" floppy discs. Measuring 315 mm by 190 mm x 77 mm, the Sternstat Disk Case weighs 1.8 kilos and retails at £32.20. It seem to be just the thing for people who work in the middle of the road or who have offices with heavy traffic problems.

Extensive trials have been carried out on the Sternstat Disc Case, which is impact resistant to the extent where a car can drive over it without damaging its contents. The product has been selected by the Design Council. In addition the case is heat resistant to UL90V0 BS Part 7 Class 2. Made in the UK, the case comes in black with two keys. More information can be obtained from Sternstat Ltd, 9 Watkin Road, Wembley, Middlesex HA9 0XL (phone 01-900 0255/6/7).

CONTROL YOURSELF

Three new, low-cost industrial control packages are now available from Stonefield, based on the company's recently introduced 'Captain' control system. Package A (illustrated) is the most complete, and is priced at £995. It comprises an Epson HX-20 personal computer, an editor system complete with PROM blaster, a target system ready fitted with industrial input/output, comprehensive hardware and software manuals, and 10 EPROMs for further control projects.

Package B, at £645, is identical to Package A except that it does not include an Epson HX-20, and Package C, at £295, consists of the target system and a 16-way digital input/output card. Package C is intended for those users who already have the Captain editor system and entry terminal, and wish to move on to further control projects. Further details can be obtained by contacting Stonefield, Denne Parade, Horsham, West Sussex, RH12 1DL (telephone 0403 51366).

SCHOOL'S OUT

The Open Computing School has been open to all-comers on two days a week for the last year at the South Bank Polytechnic, near Waterloo, London. At the last count, 150 students were currently enrolled for one of the self-paced courses. These vary from an introduction to BASIC programming, to one of the more sophisticated information retrieval systems, or databases. Students come from large organisations like British Telecom and London Transport, as well as from small businesses. In addition many come for their own education, or to keep up with their children. Most are beginners.

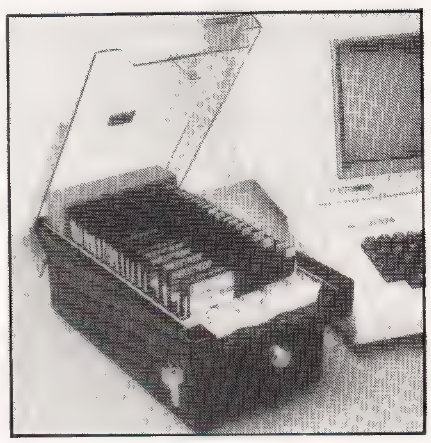
There are no timetabled classes and students can start at any time and choose when they attend according to the amount of time they wish to spend on the preparatory private study. Only time actually spent working on a microcomputer is charged for. A full course of 15 hours on the computer costs £45 for BASIC. At least one tutor is available for a maximum of eight students on the machines at once. If they wish students can be prepared for the City and Guilds Certificate in Computer Literacy.

The School has now acquired three of the Osborne small business computers as well as five BBC machines, which will be of interest to teachers who can learn about the graphics facilities, and other people who have bought then for home use. Other activities starting during the current academic year are a "serious users club" which will be half practical work and half talks and demonstrations, and a series of short courses for managers and others who want to know about the problems of specifying and running micro systems. More details can be obtained from Jack Flatau, Polytechnic of the South Bank, Borough Road, London SE1 0AA, or ring 01-928 8989 ext 2468 for a leaflet.

HANG IT ALL

Keeping track of what is on every floppy and even where that floppy is, can prove a time-consuming business. The Inmac DiskMate can neatly organise up to eighty 5¼" floppy discs in a sensible filing system designed for instant, accurate location of discs and easy retrieval.

The DiskMate comes with 20 word-processing cards and 20 data processing cards to record detailed information about each disc including space for up to two back-ups. In addition, at the front, an alphabetic set of index cards provides quick access to the right WP or DP card. Numbered self-adhesive labels are provided to identify each disc.



The discs are held in place with patent suspension clips, which fit onto the floppy's own protective Tyveck envelope and then under the file retaining bar along each side of the DiskMate. The resulting suspension filing system enables the discs to be slid back and forth along the rails for easy retrieval and a control switch will flip open the retaining bar for inserting new discs and envelopes in a matter of moments.

This compact system measures 180 mm wide by 150 mm high by 30 mm long and is lockable to deter unauthorised users. The unit is moulded from tough dark brown plastic with a smoke brown transparent hinged lid and costs £37.00. Expansion packs containing 20 WP cards, 20 DP cards, 80 filing clips and self-adhesive labels are available at £11.00 each.

Further details can be found in Inmac's free full colour catalogue along with details of nearly 1000 other products for computer users. All Inmac products carry a full one year guarantee and are available on next day delivery with a 30 day risk free trial period. Further information can be obtained from Inmac (UK) Limited, Davy Road, Astmoor, Runcorn, Cheshire WA7 1PZ (telephone 09285 67551: telex: 629819 INMAC G).

LION LINES

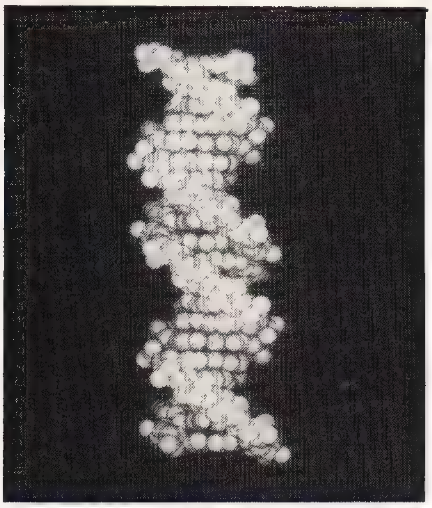
Gresham Lion (PPL) Ltd design and manufacture a wide range of technology advanced colour graphics/image display systems. These are used extensively for specialist computer display applications, for example CAD/CAM, satellite image processing, meteorology, molecular research and process control. On Stand No. 4153/4159 at Compec, November 15th-18th, Gresham will be demonstrating Supervisor 1024, their latest ultra-high-resolution, colour raster scan display system. Also on show for the first time will be the four Supervisor 214 standard configurations.

The Supervisor 1024 fully utilises the capacity of the latest generation high-resolution CRTs to provide a display format of 1024 x 768 pixels, 50 Hz non-interlaced with a physical picture area of 2048 x 2048 pixels. A totally flicker-free display has been achieved without the use of long persistence phosphors.

The system enables fast graphics generation and incorporates such advanced features as pan, zoom with smooth two-axis scroll, VDU emulation, output combination logic and colour mapping. Options include an ASCII keyboard with numeric pad, trackerball and cursor nudge keys. The system is available with a choice of powerful software. An important feature is the high update rate, 66 nanoseconds per pixel — during which the display remains totally undisturbed. A further provision is multi-page architecture enabling hidden update for display list driven applications.

The modular design, together with a variety of hardware and software options, plus a choice of local intelligence, makes it appropriate for a wide range of applications including full colour solid modelling and imaging. Interfaces include RS232, RS424, DMA, Ethernet and other Local Area Networks.

In addition to providing systems to customer specifications, there are three standard cost-effective configurations — a Colour Graphics Controller, Colour Graphics VDU and a Monochrome Pseudo Colour imaging controller. Further information can be obtained from Gresham Lion (PPL) Ltd, Lower Way, Thatcham, Berks (telephone 0635 68686).



SIRTON TO PLEASE

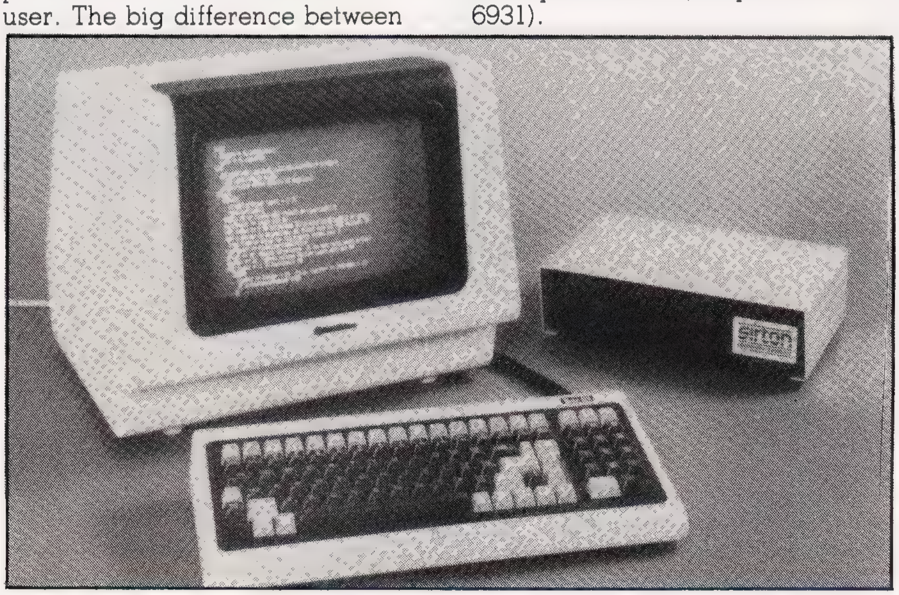
Sirton Computer Systems, manufacturers of the MIDAS range of professional microcomputers, have launched a new multi-user, multi-tasking computer system specifically for technical and scientific applications. Called the MIDAS-MPS, the system features distributed processing and can be configure for up to 16 users while maintaining CP/M compatibility.

The MIDAS MPS consists of two main sections: a central MASTER computer, and separate USER processors for each individual user. The big difference between the MIDAS MPS system and most other current multi-user packages is that, because of the master/ user arrangment, each user has a local processor at his terminal. In practice this gives a much faster access times, as well as more flexibility and computing capacity than several users all working from the same central processor. These facilities are more or less essential in scientific establishments and laboratories where computers are worked hard, and many terminals may be used simultaneously.

The master processor has mass storage facilities including removable floppy discs, large capacity high speed hard discs — up to 80M Bytes — and a separate memory with the sole task of servicing the requests of the user processors. This means that the machine works as a true master by using its own operating system at all times.

Each user module can be used as a stand alone unit — accessing the master for a few seconds only to down-load programs and data at the beginning and end of a working session — and any faults or problems arising in one user module will have no affect on the rest of the system. Each USER can have its own printer attached, or data can be printed out centrally from the master.

Typically the cost of a five user system, all with separate processors and terminals, is around £10,000. Up to 16 screens can be added to the system. Sirton Computers specialise in producing computer systems for scientific and technical establishments, and currently have more than 350 systems in use throughout the country. Further details from Sirton Computer Systems Ltd, Unit 14, 29 Willow Lane, Mitcham, Surrey CR4 4NA (telephone 01-640 6931).



NEXT MONTH

computing today

JANUARY ISSUE ON SALE DECEMBER 9th

APPLE MUSIC

Nothing to do with the famous Beatles company: this is much more versatile. From contempory tunes to classical symphonies, these machine code routines allow complex tunes to be played on your computer, provided you supply all the right data of course! And because the Apple is a 6502-based machine, it shouldn't be difficult for the owners of other machines that use this microprocessor to adapt the software for their own purposes. Get *Computing Today* next month — it's the perfect accompaniment for your computer.

ELECTRON PARTICULARS

Acorn's new baby gets put on the test bench next month. Is it a sawn-off BBC? How does it rate as a machine in its own right? Should other sub-£200 micro manufacturers break out the tranquilisers? As usual we've pulled the computer to pieces and put it back together again looking for the answers, and you can read what we think in the January issue of CT.

Articles described here are in an advanced state of preparation but circumstances may dictate changes to the final contents.

NAMED FILES FOR THE NASCOM

The NAS-SYS monitor doesn't allow tape files to have names, even a single character, which can be inconvenient at the best of times. Next month we publish a program that modifies the tape routines to improve matters out of all recognition: files can be stored on tape with headers 16 characters long. That should give you plenty of scope for filenames!

TRS-80 SCREEN EDITOR

Line editors are a bit of a pain in the keyboard. A throwback to the glorious days of teletype, they do have the advantage that an errant programmer can only mess up one line by accident, but this is offset by the inconvenience of remembering all the editing commands. In the next issue of *Computing Today* we'll be publishing some software that can transform programming on the TRS-80 by providing a full screen editor. You've never had it so good.

COMPETITION NUMBER 2

We hope a lot of you will enter the competition in this issue to devise and program an Adventure game. However, many won't. Perhaps your programming isn't up to scratch, or you lack a decent computer system. The only remedy for the former problem is to keep reading *Computing Today,* but next month we'll have a competition with a prize that will solve the hardware shortage for one lucky reader. It's a little difficult to describe exactly what the prize will be — you'll see why next month — but the competition will not be unconnected with the theme of Adventures. Don't miss the January issue of *Computing Today*.

GIVE THIS TO YOUR NEWSAGENT

Please reserve me a regular copy of Computing Today

NAME ....

ADDRESS ....

POSTCODE ....

Tony Cross and Phil Cornes

MIKRO MAGIC

It's difficult to get the very best out of the Commodore 64 without resorting to machine code, and here we review a product to make that a lot easier.

When we first took delivery of our Commodore 64 we very quickly formed the impression that it wasn't really meant to run BASIC. It was obvious that Commodore had aimed it at the cartridge software market, with the result that the majority of its powerful facilities are best accessed from machine code.

For the serious home user the only real problem this presents is that he needs access to a good assembler. Unfortunately, good quality assemblers for cassette-based machines have not been readily available — at least not until Supersoft recently released a Commodore 64 version of the PET-proven MIKRO Assembler. However, at £50 plus VAT it's not cheap, so we decided to put one through its paces to find out just what you can expect for your money.

COMPACT CARTRIDGE

The MIKRO Assembler comes as a compact cartridge (80 mm x 70 mm x 20 mm) which plugs directly into the expansion port on the back of the machine. Our first complaint here — the manual didn't tell us which way round it should go and, of course, it will fit either way. We assumed that the plain side faced down, which fortunately turned out to be correct, so we've no idea what happens if you plug it in the wrong way round (and at £50 a time we're not going to try!). Once it is fitted, however, it's there 'permanently' — no messy loading of tapes, an important feature of an editor/assembler. (In our experience there is nothing worse than having the editor and the assembler as separate programs on tape because you can waste a lot of time just waiting for them to load!)

WHAT YOU GET

The MIKRO cartridge is really three 'programs' in one because you get a very useful editor, a fast three-pass assembler and a very nice little monitor as well. And because they are 'permanently resident' you can switch between them *and* the normal BASIC editor/interpreter quite freely. This compatibility between the MIKRO package and resident BASIC means that you can leave the MIKRO cartridge permanently plugged in even when you are only using BASIC. In fact, the only difference that MIKRO makes to the standard machine is that you 'lose' 8K of BASIC RAM from $8000 to $9FFF, which is where the MIKRO ROM locates itself.

THE EDITOR

The MIKRO editor is really nothing more than the standard BASIC screen editor with a few very useful additions. The source code is entered (and stored) just like a BASIC program with a line number on each line. The format and syntax of the line, however, is totally different. Each source code line consists of four 'fields' — the label, opcode, operand and comment fields. The fields must be separated by at least one space and the comment field must also be preceded by an exclamation mark (!). For example, Listing 1 shows typical MIKRO source code. Just as with BASIC, the maximum line length is 80 characters (two lines) which allows for some fairly lengthy comments — very useful in machine code!

To make the editing process easier the MIKRO package provides some very handy additional facilities. The first of these is AUTO line numbering which saves typing in a new line number each time. For example, AUTO 10,10 generates line numbers which start at 10 and increment by 10 each time. We found this to be very useful because you don't need to interrupt your 'train of thought' each time to type in a new line number.

Equally useful is a block DELETE command which allows you to delete several contiguous lines of code all at the same time. For example DELETE 110-250 would delete all lines of code from 110 to 250 inclusive.

Almost as important for machine code work is the FIND command which finds all occurrences of a given string. For example, FIND LDA #$3 would find and print on the screen all the occurrences of the string 'LDA #$3'.

One small problem with the

editor is that the source code can be difficult to read, particularly if your labels are different lengths. To overcome this problem MIKRO has a FORMAT command which lists the code in a more readable format. For example, Fig. 1 shows MIKRO source code listed with a LIST command and with a FORMAT command.

```
100 BEGIN  LDY #$5
110        LDA #CHAR
120 STLOOP STA PTR,Y
130        INY
140        BNE STLOOP
150        RTS
```

```
100 BEGIN LDY #$5
110 LDA #CHAR
120 STLOOP STA PTR,Y
130 INY
140 BNE STLOOP
150 RTS
```

Fig. 1. The FORMAT command (top), compared to the normal LIST command (bottom).

In case you have difficulty converting from decimal to hex or binary, MIKRO has a NUMBER command which converts any number in the range 0-65535 decimal to its hex, octal and binary equivalents. For example, a NUMBER 42 command would produce the following output:

```
$002A  42  @000052  %0000000000101010
```

where the '$' prefix indicates a hex number, the '@' prefix indicates an octal number, the '%' prefix indicates a binary number and no prefix indicates a decimal number.

The MIKRO source code can be SAVEd, LOADed and VERIFYd just like any other BASIC program, and as an added bonus the additional MIKRO facilities can be used when writing normal BASIC programs (although FORMAT and NUMBER have limited uses!).

One facility which we would have liked but didn't find is a line renumber facility. In our experience it's quite common when debugging machine code programs to have to insert several new lines of code between two existing lines. Without a renumber facility, once all the line numbers have been used up it is no longer possible to do this. (Fortunately, a simple renumber routine is fairly easy to write because there are no GOTOs or GOSUBs to complicate matters.)

THE ASSEMBLER

The MIKRO assembler is a three-pass assembler, which means that it 'scans' the source code three times to produce the correct object code. The main advantage of multi-pass assemblers is that they don't impose restrictions on the use of forward label references. For example, most single-pass assemblers would not be able to assemble the source code in Listing 2 because of the forward reference to the label 'TEST'. Multi-pass assemblers, like MIKRO, would have no difficulty in assembling this code.

There is one restriction which MIKRO does impose, however, and that is on the use of multiple forward references. For example, MIKRO cannot assemble the source code in Listing 3 because of the multiple forward reference necessary to find the value of 'VARPTR'. This doesn't really present any problems, it just means that a little thought is required when defining 'variables' and 'equates'.

The main disadvantage of multi-pass assemblers is that the extra passes can sometimes slow them down. This is certainly not the case with MIKRO — it is very, very fast. As an experiment we assembled around 14K of source code, which was about half code and half comments, and MIKRO took just 7 seconds to produce 1102 bytes of object code! Supersoft's claim that MIKRO can assemble 4K of object code in 20 seconds is probably quite true.

Apart from its speed we were also impressed by the range of pseudo-ops which are available. (Pseudo-ops are instructions to the assembler which are entered into the source code just like 6502/6510 instructions.) As well as the standard ones like '=' (which assigns a value to a label) and 'BYT' (which loads a byte of memory with a value), MIKRO also suports 'WOR' which loads two bytes of memory with a 16-bit value and 'TXT', which is a sort of multiple 'BYT' instruction used for storing text strings. In all of these pseudo-ops, except 'TXT', any of the four standard number bases can be used (hex, decimal, octal and binary). In addition ASCII characters can be entered by prefixing them with the ' character. For example, the source code line

```
100 BYT $41,65,@101,%0000000001000001,'A
```

would assemble into five consecutive bytes all with the value 65 decimal.

Most assemblers support the pseudo-ops '*' which means 'use the current assembly address' and '*=' which means 'load the address counter with the following value'. For example:

```
100 *=$C000
```

means start assembly at address $C000, while:

```
110 FRED   = *
```

means assign the current assembly address to the label FRED. In this case FRED will equal $C000.

In addition MIKRO supports the following special form of the '*=' pseudo-op:

```
100 *=$A000,$C000
```

This means 'assemble the code to run at address $A000 but locate it at address $C000'. We found this facility very useful for producing code which will eventually run at an address currently overlaid by ROM (in this case the BASIC ROM), but which needs to be assembled into RAM so that it can be debugged or saved onto tape.

Two further special pseudo-ops which MIKRO supports are the 'LNK' and 'END' instructions. These are used when the source code needs to be split into separate tape files because it is too big to fit in RAM. If it becomes necessary to split a program then the 'LNK' instruction is used to link the files together. For example, imagine that a long program has been split into two files called "FILE ONE" and "FILE TWO". The 'LNK' instruction must be used at the end of "FILE ONE" to link in "FILE TWO" as follows:

```
3450        LNK "FILE TWO"
```

The 'END' instruction must be used at the end of "FILE TWO" to link back to "FILE ONE" as follows:

```
2960        END "FILE ONE"
```

The 'END' instruction is needed by the assembler so that it can tell when it has read all the files.

You can link as many files as you like as long as the object code generated is less than 12K, which is the most that MIKRO can handle in one go. Longer programs would have to be assembled separately and linked 'by hand' afterwards.

When you are linking files in this way all of the files need to be loaded in three times, once for each pass of the assembler. While this takes a long time (tape loading time, not assembly time), MIKRO makes the operation as painless as possible by prompting for the next file it requires.

THE MONITOR

The main functions of the MIKRO monitor are to enable machine code programs to be debugged, saved and loaded. Saving and loading machine code is achieved by simply copying the appropriate area of memory to the cassette, or vice versa. For example, to save the area of memory from $C000 to $C44C you use the following command:

```
.S "PROGRAM",01,C000,C44D
```

where "PROGRAM" is the file name, 01 is the device number (the cassette), C000 is the start address and C44D is the end address (both in hex). To load the same program you use the following command:

```
.L "PROGRAM"
```

The start address is contained in the file header on the tape, so you don't need to worry about it.

For debugging programs you need to be able to operate on the object code in memory. The monitor has a number of useful facilities to help you to do this. For example, the modify command (.M) will display the contents of any specified area of memory: you can change the values in these memory locations by simply overtyping them. We found an annoying problem with this command, though. If you ask it to display an area of memory which is longer than the screen, it just keeps scrolling until the end address is reached. It would be so much nicer if it had some sort of 'pause' feature, say pressing the space bar, so that you could interrupt the listing to read it bit by bit.

For searching for specific bit patterns MIKRO has a hunt command (.H) which lists all the occurrences of the particular bit pattern. For example, .H C000 C44C 8D 61 42 will list the address of every occurrence of '8D 61 42' between $C000 and $C44C. This would be handy if, say, you had to change the value of an 'equate', because you can hunt for every occurrence of the old value to make sure that they all get changed.

If you have used the '*= address, address' pseudo-op during assembly you will eventually want to copy the program from its assembled address into its execute address. This can be done by using the transfer command (.T) which will copy any area of memory into any other. For example, .T C000 C44D A000 will copy the block of memory from $C000-$C44C to $A000-$A44C.

Machine code programs can also be executed from the monitor by using the goto command (.G). For example, .G C000 would start executing a program at address $C000.

By far the two most useful facilities for debugging programs are the ability to set breakpoints and to single-step through the code. MIKRO has no special facilities for setting breakpoints but they can be entered by using the .M command and writing them in. (A breakpoint is just the 6502/6510 BRK instruction, which assembles to $00). When a breakpoint is encountered during execution, control is passed back to the monitor which displays the contents of all the registers. These can be altered by simply overtyping, and you can use any of the other monitor facilities as well. To continue execution you now have to replace the original instruction at the breakpoint address and then execute from there by using the .G command.

A single-step mode would allow you to execute one instruction at a time without having to keep inserting/removing breakpoints. Sadly MIKRO does not possess such a facility. This is a pity because single step makes debugging so much easier — doing it by inserting a breakpoint 'manually' every time is very time-consuming.

As a final aid to debugging the monitor has a disassemble command (.D) which will print a disassembled listing of any area of memory. Unlike the .M command, the .D command pauses at the end of each page — pressing any key prints the next page. This facility is particularly useful because it displays the source and object codes together, so that you can quickly see how the object code in memory relates to your original source code.

CONCLUSION

Overall we were very impressed with the MIKRO package. It is both fast and easy to use, and it works equally well with both disc and cassette-based systems. The only real faults we could find with it are the lack of a line renumber facility in the editor and the lack of a single-step facility in the monitor.

As to whether it's 'value for money' we give it a unanimous 'YES'. After all, £50 is really quite a small price to pay for software of this quality. (We have used professional assemblers costing three time the price which don't perform any better.) Further details on the MIKRO Assembler can be obtained from Supersoft, Winchester House, Canning Road, Wealdstone, Harrow HA3 7SJ (telephone 01-861 1166).

```
100 SCREEN = $0400
110 START LDA #$41 !THE PROG START
120 LDY #$0
130 LOOP STA (SCREEN),Y !PUT 'A' ON SCREEN
140 INY
150 BNE LOOP !REPEAT 256 TIMES
160 RTS !FINISHED
```

Listing 1. Typical MIKRO source code.

```
100          LDA #TEST
110          STA TSTCHR
120          BNE LOOP
130 TEST     = $43
```

Listing 2. MIKRO can cope with this forward label...

```
100          LDA #VARPTR
110          STA TSTCHR
120          BNE LOOP
130 VARPTR = TEST
140 TEST     = $43
```

Listing 3. ...but not with this.

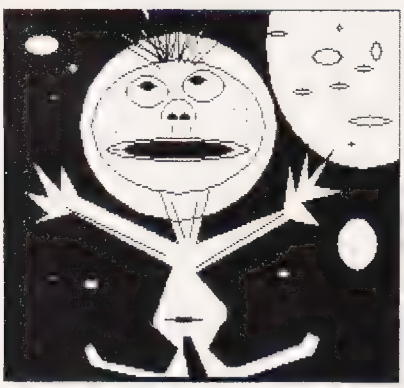
David Mitchell

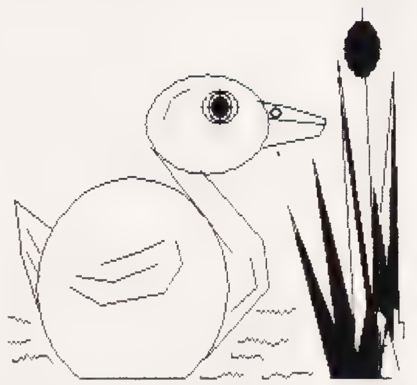
DRAGON GRAPHICKS

Owners of the Dragon 32 no longer need paint and canvas: exercise your artistic tendencies with this package of screen drawing commands.

Although the Dragon 32 has very advanced graphic capabilities, programmers fail to make full use of them due to the difficulty of calculating numerous plotting coordinates. Even very simple pictures require a large number of program lines and careful planning with the aid of a pencil and graph paper. More advanced applications, such as the drawing of a map outline, present such a daunting task that most mortals will just not bother. Graphics tablets which allow direct input of freehand drawings are one answer to the problem but these are very expensive and as yet unavailable for the Dragon. Graphic tracers which plug into the joystick port are a cheaper alternative but although produced for the BBC and the Spectrum, have not yet appeared for the Dragon.

This program presents the solution in software and allows the user to rapidly produce high quality graphics for use in other programs or for printing using an Epson printer (optional). The main program is written in BASIC and a small machine code routine is used to dump the screen contents to a printer. (The dump program is loaded separately so that it may be used elsewhere.) The program requires one joystick which should be plugged into the right joystick socket.





WHAT IT DOES

On running the 'Graphicks' program the high resolution screen appears blank except for a small flashing cursor accompanied by a ticking noise. (If you find this irritating after a while, turn the volume down on your television!) You may feel lost at this point, so pressing 'H' for help reveals the instruction page. The instructions consist of a list of single key commands:

"C" = circle
"L" = line
"B" = rectangle
"K" = solid block
"P" = paint
"E" = erase
"F" = freehand draw
"G" = 'get' an area of picture
"R" = repeat draw
"T" = turtle graphics
"S" = save to tape
"#" = dump screen to printer
"!" = clear the screen
Pressing "H" at any time returns you to the instruction page.

To use "Graphicks", simply select the type of shape you wish to draw by pressing the single key required, move the cursor to the position on the screen where you want to draw it and press the fire button on the joystick. This fixes the origin of the shape; now move the joystick around to alter the size and position of the shape. Once you have decided that it is correct press the fire button again; the shape is now fixed on the screen and the cursor returns, once more under the control of the joystick. Now for the clever bit; press the fire button again and move the joystick around so that the new shape overwrites the first one. Notice how the original is unaffected by the second; this allows you to try out additions to your masterpiece without destroying it with an inadvertant sllip of the joystick. Only by pressing the fire button again does a permanent second shape appear on the screen.

The line option (the default mode) allows the user to draw straight lines between the selected origin and any other point on the screen. This type of effect is known as rubber banding.

The rectangle and filled block options operate in a similar fashion, having one corner fixed and the size and shape determined by the joystick position.

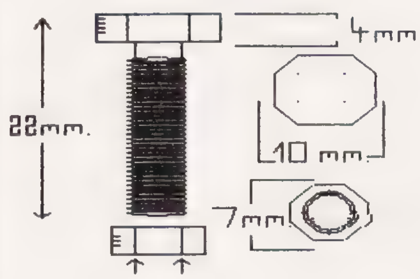
The circle routine is slightly different, the cursor fixes the central point and the x joystick direction controls the radius of the circle. The y joystick direction controls the height/width ratio. This greatly simplifies the drawing of ellipses and the rubber banding makes it easier to position them accurately.

The erase option draws a filled block in the background colour; this allows the deletion of partial areas of the screen and can also be used to produce white blocks on black backgrounds.

Paint does just that! Select the area needing to be painted by using the joystick and press the fire button. The area fills with colour — then empties! Fortunately it continues this cycle until you decide to accept its handiwork and press the fire button. Don't panic if you've left a hole in the selected area and the 'paint' is running all over your Van Gogh — just press another menu option and you're back to where you started.

Listing 1. The 'Graphicks' program.

```
10  ****************************
20    GRAFFICKS
30
40    BY DAVID MITCHELL
50
60   a graphics utility
70 ' for the dragon 32
80 '
90 '*****************************
100 CLEAR 500,31999
101 PCLEAR8
102 DIML(16):DIMG(30.30)
105 GOSUB 15000
110 PRINT@L(15),"..........LOADING MACHINE CODE"::CLOADM 'DUMP"
120 POKE65495,0
130 PMODE4,5:COLOR0,5:PCLS
140 PMODE4,1:COLOR0.5:PCLS
150 M$="LCBKPEFTHSGR"
160 T$="T15L1601C"
165 TR$="T6001CEG02CEG03CEG04CEG05CEG"
170 SCREEN1,1
180 PK=65280
190 P=1
200 GOSUB 10000
1000 '**********READ JOYSTICKS
1010 X=4*JOYSTK(0)
1020 Y=3*JOYSTK(1)
1030 RETURN
2000 '**********READ FIRE BUTTON
2010 IF PEEK(PK)=126 OR PEEK(PK)=254 THEN F=1:ELSE F=0
2030 RETURN
3000 '**********READ KEYBOARD
3010 A$=INKEY$
3015 IF A$="S" THEN GOSUB 11000:RETURN
3016 IF A$="H" THEN GOSUB 12000:RETURN
3017 IF A$="!" THEN PCLS:P=1:RETURN
3018 IF A$="#" THEN EXEC:RETURN
3020 IF A$="" THEN P=P:ELSE P=INSTR(1,M$,A$)
3030 RETURN
4000 '**********POSITION CURSOR
4010 C=PPOINT(X,Y)
4020 PSET(X,Y,0)
4030 PLAY T$
4040 PRESET(X,Y)
4050 PLAY T$
4060 PSET(X,Y,C)
4070 RETURN
5000 '**********FIX POINT
5010 IF F=0 THEN RETURN
5020 F=0
5030 SOUND25,1
5045 GOSUB 6000
5050 RETURN
6000 '**********ACTION
6005 A=X:B=Y
6006 GOSUB 8000
6010 GOSUB 1000
6020 GOSUB 2000
6030 GOSUB 3000
6040 ON P GOSUB 7000,7100,7200,7300,7400,7500,7600,7700,11000,12000,13000,14
6045 GOSUB2000
6050 IF F=1 THEN F=0:PLAY TR$:RETURN
6055 IF P>6 AND P<9 THEN 6070
6060 GOSUB 9000
6070 GOTO6010
6080 RETURN
7000 '**********LINE DRAW
7010 LINE(A,B)-(X,Y),PSET
7020 RETURN
7100 'CIRCLE DRAW
7110 Y=10/(Y+1):X=X/2
7115 TS=Y
7120 CIRCLE(A,B),X,,Y
7130 RETURN
7200 'BLOCK DRAW
7210 LINE(A,B)-(X,Y),PSET,B
7220 RETURN
7300 'FILLED BLOCK DRAW
7310 LINE(A,B)-(X,Y),PSET,BF
7320 RETURN
7400 'PAINT
7410 PAINT(X+1,Y),0,0
7415 FORT=1 TO 50:NEXTT
7420 RETURN
7500 'ERASE
7510 LINE(A,B)-(X,Y),PRESET,BF
7520 RETURN
7600 'FREEHAND DRAW
7610 LINE(A,B)-(X,Y),PSET
7620 A=X:B=Y
7630 RETURN
7700 'TURTLE GRAPHICS
7705 IF X<20 OR X>230 THEN X=SGN(X-128):ELSE X=0
7710 IF Y<20 OR Y>170 THEN Y=SGN(Y-96):ELSE Y=0
7720 IFA+X>256 OR A+X<0 THEN X=0
7730 IF B+Y>192 OR B+Y<0 THEN Y=0
7740 PSET(A+X,B+Y,0)
7750 A=A+X:B=B+Y
7760 RETURN
8000 '**********COPY 1-2
8010 FOR I=1 TO 4
8020 PCOPY I TO I+4
8030 NEXT
8040 RETURN
9000 '**********COPY 2-1
9010 FOR I=1 TO 4
9020 PCOPY I+4 TO I
9030 NEXT
9040 RETURN
10000 '*********MAIN PROG
10010 GOSUB 1000
10020 GOSUB 2000
10030 GOSUB 3000
10040 GOSUB 4000
10050 GOSUB 5000
10060 GOTO10000
10070 RETURN
11000 '*********SAVE SCREEN TO TAPE
11002 POKE65494,0
11010 CLS
11020 PRINT@L(1),"*********************************";
11025 PRINT@L(2)+11,"graffícks";
11026 PRINT@L(3),"*********************************";
11030 PRINT@L(5),"save diagram to tape";
11040 PRINT@L(8),"enter file name";
11045 SCREEN0,1
11050 INPUT F$:PRINT"";
11060 IF F$="" OR LEN(F$)>8 THEN11010
11070 PRINT@L(12),"set up tape to record";
11075 PRINT@L(8),"file name:-........     ";F$;
```



The freehand draw option turns the joystick into a pencil. You simply press the fire button once you have selected your starting point and draw directly on the screen. Again don't worry if you make a mistake, simply press 'L' to return you to the line mode (this wipes out your error) then return the cursor to its original position, press 'F' and start again!

The turtle graphics work in the same way except that the joystick controls a moving line on the screen. There are eight possible directions and moving the joystick to a central position stops the movement. As previously, nothing is permanent until the fire button is pressed again.

To repeat small patterns on the same screen a facility exists for selecting an area of the picture and then placing it as many times as required anywhere else on the screen. The repeat pattern is retained in memory and can be used at any time until changed by use of the select mode, or until the screen is cleared.

Once completed you may wish to save your drawing on tape (eg for use as a background in a games program). To do this press 'S'. You will be given the opportunity to enter a filename and prompted to set up the tape recorder (hands up everyone who has saved a program to an empty tape recorder!!). The diagram is saved as a machine code program which can then be loaded from within a BASIC program. Try drawing a picture and saving it on tape, then switch off your computer and switch it back on again. Rewind the tape and type in the following short program:

```
10 PMODE4,1:COLOR 0,5:PCLS:SCREEN1,1
20 CLOADM F$     (where F$ is the file
name of your picture)
30 GOTO 30
```

and then type RUN and switch on your tape recorder to PLAY. Your saved diagram/drawing should reappear before your eyes. In games programs more than one drawing can be recorded after the main program to be loaded as required. The loading of the first screen could be conveniently timed to coincide with the display of instructions on the low

```
11080 PRINT@L(13),"press any key when ready";
11081 SCREEN0,1
11082 A$=INKEY$:IF A$="" THEN 11082
11085 SCREEN0,1
11090 PRINT@L(12),"please wait...........";
11095 PRINT@L(13),"        recording.....  ";F$;
11100 SCREEN0,1
11110 CSAVEMF$,1536,7679,1536
11120 SCREEN1,1
11125 POKE65495,0
11130 RETURN
12000 '*********SUMMARY OF COMMANDS
12005 CLS
12010 PRINT@L(1)+11,"gràfficks";
12020 PRINT@L(3)+3,"l= LINE DRAW";
12030 PRINT@L(4)+3,"b= RECTANGLE DRAW";
12040 PRINT@L(5)+3,"k= SOLID BLOCK DRAW";
12045 P=1
12050 PRINT@L(6)+3,"f= FREEHAND DRAW";
12060 PRINT@L(7)+3,"t= TURTLE GRAPHICS";
12070 PRINT@L(8)+3,"e= ERASE PART OF SCREEN";
12075 PRINT@L(9)+3,"!= CLEAR SCREEN";
12080 PRINT@L(9)+3,"g= GET SCREEN BLOCK";
12090 PRINT@L(10)+3,"r= REPLACE SCREEN BLOCK";
12100 PRINT@L(11),"--------------------------------";
12110 PRINT@L(12)+3,"s= SAVE PICTURE TO TAPE";
12120 PRINT@L(13)+3,"#= DUMP SCREEN TO PRINTER";
12130 PRINT@L(14)+3,"!= CLEAR SCREEN";
12140 PRINT@L(15),"********************************";
12150 PRINT@L(16),"  press any key to continue";
12155 SCREEN0,1
12160 A$=INKEY$:IF A$="" THEN 12160
12165 SCREEN1,1
12170 RETURN
12999 GOTO12999
13000 '*********GET SCREEN BLOCK
13010 IF X>225 THEN X=225
13020 IF Y>161 THEN Y=161
13030 GET(X,Y)-(X+30,Y+30),G,G
13040 PUT(X,Y)-(X+30,Y+30),G,PRESET
13050 GOSUB 9000
13060 RETURN
14000 '*********PUT SCREEN BLOCK
14010 IF X>225 THENX=225
14020 IF Y>161 THEN Y=161
14030 PUT(X,Y)-(X+30,Y+30),G,AND
14050 RETURN
15000 '*********INTRO PAGE****
15005 FORI=1TO16:L(I)=(32*I)-32:NEXT
15010 CLS2
15020 IP$=STRING$(31,202)
15030 PRINT@L(1),IP$;
15050 PRINT@L(6)+11,"graffícks";
15060 PRINT@L(9)+5,"A DRAWING UTILITY FOR THE";
15070 PRINT@L(10)+13,"DRAGON 32";
15080 PRINT@L(13)+6,"please wait......";
15090 PRINT@L(16),IP$;
15100 IF PEEK(&H7D00)=&H34 THEN FORI=1 TO 3000:NEXTI:GOTO 120
15110 RETURN
```

Listing 2. The BASIC loader for the screen dump program. Only lines 310 onwards need be entered.

```
10 '******* DUMP ***************
20 '
30 ' by david mitchell
40 '
50 ' a machine code routine to
60 ' dump the contents of the
70 ' DRAGON 32 high resolution
80 ' screen to an EPSON FX-80
90 ' printer
100 '
110 ' note:- this dump works in
120 '        both PMODE3 and
130 '        PMODE4
140 '
150 ' Once loaded and run this
160 ' program should be saved
170 ' by typing :-
180 '
190 'CSAVEM"DUMP",&H7D00,&H7D4E,&H7D00
200 ' the program is then loaded
210 ' by a CLOADM command and run
220 ' by typing EXEC.
230 '
240 ' If used with the GRAFFICKS
250 ' program it should be saved
260 ' on the same tape directly
270 ' after the main program.
280 '
290 '****************************
300 '***** ENTER PROGRAM FROM HERE
310 FORI=&H7D00 TO &H7D4D
320 READ V$
330 POKEI,VAL("&H"+V$)
340 V=V+PEEK(I)
350 NEXT I
360 IF V<>7864 THEN CLS:PRINT"ERROR IN DATA ENTRY":STOP
370 CLS
380 PRINT@64,"DATA ENTERED CORRECTLY"
390 PRINT@96,"----------------------";
400 DATA34,36,8E,1D,E0,86,1B,BD,80,0F,86,33,BD,80,0F,86,17,BD,80,0F,86,1B,BD,80,
0F
410 DATA 86,2A,BD,80,0F,86,00,BD,80,0F,86,C0,BD,80,0F,86,00,BD,80,0F,C6,C0,86,84
,43
420 DATA BD,80,0F,30,88,E0,5A,C1,00,26,F2,86,0D,BD,80,0F,30,89,18,01,8C,1E,00,26
,BA,35,36,39
```

Listing 3. The machine code dump routine corresponding to the DATA in listing 2.

```
  40      PRT                         40      FML
4001
7D00                                 140      ORG   $7D00
800F                                 150 @PRINT   EQU   $800F
7D00 3436                            160 @START   PSHS  A,B,X,Y
7D02 8E1DE0                          170      LDX   #$1DE0
7D05 861B                            230 @LINE  LDA   #$1B
7D07 BD800F                          240      JSR   @PRINT
7D0A 8633                            250      LDA   #$33
7D0C BD800F                          260      JSR   @PRINT
7D0F 8617                            270      LDA   #$17
7D11 BD800F                          280      JSR   @PRINT
7D14 861B                            290      LDA   #$1B
7D16 BD800F                          300      JSR   @PRINT
7D19 862A                            310      LDA   #$2A
7D1B BD800F                          320      JSR   @PRINT
7D1E 8600                            330      LDA   #$00
7D20 BD800F                          340      JSR   @PRINT
7D23 86C0                            350      LDA   #$C0
7D25 BD800F                          360      JSR   @PRINT
7D28 8600                            370      LDA   #$00
7D2A BD800F                          380      JSR   @PRINT
7D2D C6C0                            430      LDB   #192
7D2F A684                            440 @LOOP  LDA   ,X
7D31 43                              450      COMA
7D32 BD800F                          460      JSR   @PRINT
7D35 3088E0                          470      LEAX  -32,X
7D38 5A                              480      DECB
7D39 C100                            490      CMPB  #$00
7D3B 26F2                            500      BNE   @LOOP
7D3D 860D                            510      LDA   #$0D
7D3F BD800F                          520      JSR   @PRINT
7D42 30891801                        530      LEAX  $1801,X
7D46 8C1E00                          540      CMPX  #$1E00
7D49 26BA                            550      BNE   @LINE
7D4B 3536                            560      PULS  A,B,X,Y
7D4D 39                              570      RTS
7D4E                                 580      END   @START
```

resolution screen. By adding an offset of 6143 to the CLOADM instruction two pictures can be stored at once and selected using the SCREEN command. (Remember to PCLEAR 8 before trying this, to reserve sufficient space.)

HOW IT WORKS

Lines 100-200 Clears memory for machine code and eight pages of video memory. Sets up the screen mode and defines strings used for sound and menu selection.
Lines 1000-1030 Routine for reading the joysticks.
Lines 2000-2030 Routine for testing for fire button press.
Lines 3000-3020 Scans keyboard for input, and changes the action variable (P) if requested. Also branches to Help page and Save routine if the relevant keys are pressed.
Lines 4000-4070 Sets the cursor position and flashes the cursor until the fire button is pressed. Also restores the point on the screen to the original colour once the cursor has moved away.
Lines 5000-5050 Fixes the cursor position in response to the fire button. Also makes a distinctive if not head-splitting noise: change line 165 if you can't stand it.
Lines 6000-6080 The guts of the program: how it works goes something like this. First it copies video pages 1-4 onto pages 5-8; this saves the original state of the screen while you're mucking about. It then reads the joysticks and the fire button and then scans the keyboard. Assuming you haven't pressed anything it will go to the line draw subroutine (the default mode) and draw a line on the screen between the fixed cursor and the current x,y position of the joystick. Again, if you haven't pressed the fire button, it will then redraw the original screen stored in video pages 5-8 back onto pages 1-4. It continues this loop, drawing lines as you wish, until you press the fire button again, upon which it omits the redrawing of the original screen and instead copies the new screen (1-4) onto the old one (5-8). If at any stage you select another drawing option, line 6040 directs the drawing part of the program to the relevant subroutine.

Lines 7000-7630 Subroutines to do the actual drawing.
Lines 8000-8040 The page copy routine (1-4 to 5-8).
Lines 9000-9040 The page copy routine (5-8 to 1-4).
Lines 10000-10070 Main program.
Lines 11000-11130 Tape save routine.
Lines 12000-12999 Prints the help page.
Lines 13000-13999 Copy a selected area of screen (30 by 30 pixels) into a 'GET' array.
Lines 14000-14999 'PUT's array defined in 13000 back onto the screen at current joystick position.

CONVERSION

As it stands the program should run on a Tandy Color computer if you omit the screen dump routine. It should convert quite easily to any other machine with high-resolution graphics and the ability to support two screens at once, providing that there is a quick screen copy routine in BASIC. Only the action subroutines would need major surgery. The rest should be re-written!

VARIABLE	FUNCTION
L(1-16)	Array used to position details on text pages
M$	String containing all the letters used in menu selection
T$ and TR$	Strings responsible for the sounds
PK	Address of the fire button input
P	The 'action variable': 1=line, 2=circle and so on
X,Y	The joystick position
A,B	The fixed cursor position
A$	The key input (INKEY$)
C	The colour of the point occupied by the cursor
F	State of the fire button (1=pressed, 0=unpressed)
F$	The name of the file to be saved

Table 1. The functions of the main variables used by the program.

SNOWBALL

at £9.90 is a great new adventure for:

BBC 32K ***COMMODORE 64*** ***SPECTRUM*** 48K

LYNX 48K ***NASCOM*** 32K ***ORIC*** 48K ***ATARI 400/800*** 32K

Snowball is a massive adventure with over 7000 locations. it took nine months to perfect and marks a new leap forward in adventure games – it has a detailed, planned background and is set aboard a huge starship that would really work. Snowball could be a glimpse of the future!

You play Kim Kimberley, security agent. Your mission is to guard the colony ship Snowball 9 from sabotage.

Thus when your freezer-coffin wakes you with the Snowball still in flight, you know that something must be very wrong. You're weakened and disorientated by lengthy hibernation, but the fate of the 5 mile long space-ship is in your hands!

Snowball is our new fourth adventure. Here's what the reviewers have just said about the first three:

"The Level 9 Adventures are superbly designed and programmed, the contents first rate. The implementation of Colossal Cave is nothing short of brilliant; rush out and buy it. While you're at it, buy their others too. Simply smashing!"

- SOFT, September 83

"Of the programs reviewed here, the only one that is wholly admirable is Level 9's Colossal Adventure."

- Your Computer, September 83

"I found Dungeon exceedingly well planned and wriiten, with a fast response. There are well over 200 locations and the descriptions are both lengthy and interesting."

- Computer & Video Games, September 83

"This has to be the bargain of the year . . . If adventures are your game, then this is your adventure".

- Home Computing Weekly, 30th August 83

"There are three excellent adventures on offer from Level 9 . . . the descriptions are so good that few players could fail to be ensnared by the realism of the mythical worlds where they are the hero or heroine . . . great fun to play."

- Which Micro?, August 83

MIDDLE EARTH ADVENTURES

for the same micros as Snowball

Each of these games has over 200 locations and a host of puzzles. They can be played singly or together as an impressive trilogy. Each game could well take months to solve!

1) Colossal Adventure
The classical mainframe game "Adventure" with all the original puzzles plus 70 extra rooms.

2) Adventure Quest
An epic puzzle journey in Middle Earth..

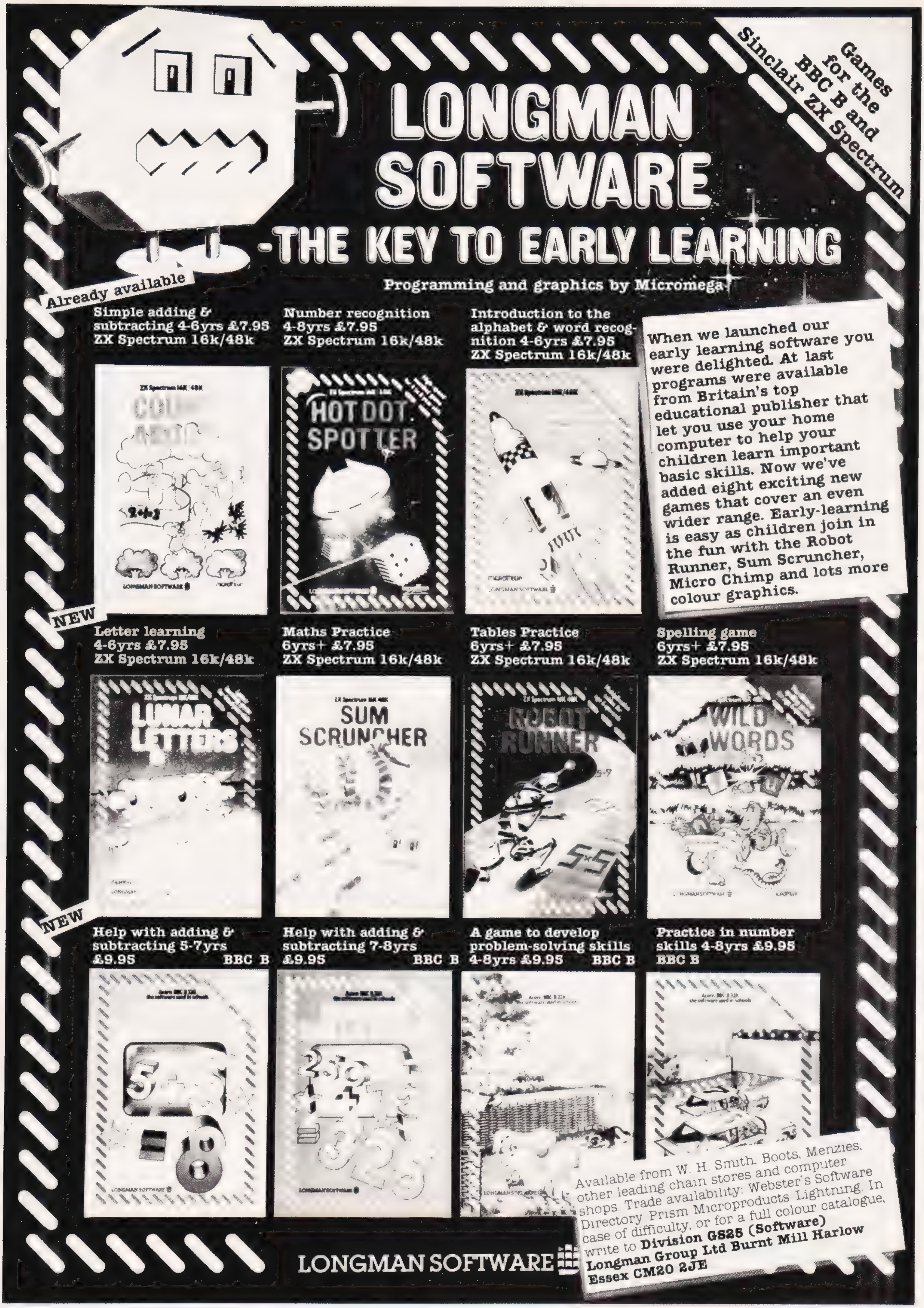
3) Dungeon Adventure
Over 100 puzzles to solve as you explore the dungeons left by the Demon Lord – outwitting their guardians.

Price: £9.90 each (inclusive)
Level 9 adventures are available from good computer shops, or mail-order from us at no extra charge. Please send order, or SAE for catalogue, describing your micro to:

LEVEL 9 COMPUTING

Dept C, 229 Hughenden Road,
High Wycombe, Bucks HP13 5PG

Dealer Enquiries Welcome

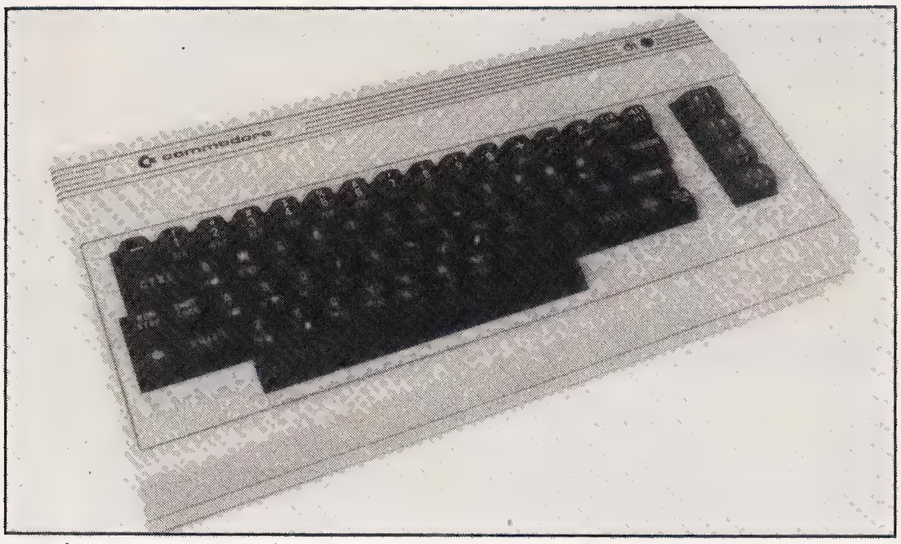


Games for the BBC B and Sinclair ZX Spectrum
LONGMAN SOFTWARE
-THE KEY TO EARLY LEARNING
Programming and graphics by Micromega
Already available
Simple adding & subtracting 4-6yrs £7.95 ZX Spectrum 16k/48k
Number recognition 4-8yrs £7.95 ZX Spectrum 16k/48k
HOT DOT SPOTTER
Introduction to the alphabet & word recognition 4-6yrs £7.95 ZX Spectrum 16k/48k
When we launched our early learning software you were delighted. At last programs were available from Britain's top educational publisher that let you use your home computer to help your children learn important basic skills. Now we've added eight exciting new games that cover an even wider range. Early-learning is easy as children join in the fun with the Robot Runner, Sum Scruncher, Micro Chimp and lots more colour graphics.
NEW
Letter learning 4-6yrs £7.95 ZX Spectrum 16k/48k
LUNAR LETTERS
Maths Practice 6yrs+ £7.95 ZX Spectrum 16k/48k
SUM SCRUNCHER
Tables Practice 6yrs+ £7.95 ZX Spectrum 16k/48k
ROBOT RUNNER
Spelling game 6yrs+ £7.95 ZX Spectrum 16k/48k
WILD WORDS
NEW
Help with adding & subtracting 5-7yrs £9.95 BBC B
Help with adding & subtracting 7-8yrs £9.95 BBC B
A game to develop problem-solving skills 4-8yrs £9.95 BBC B
Practice in number skills 4-8yrs £9.95 BBC B
Available from W. H. Smith, Boots, Menzies, other leading chain stores and computer shops. Trade availability: Webster's Software Directory Prism Microproducts Lightning. In case of difficulty, or for a full colour catalogue, write to Division GS25 (Software) Longman Group Ltd Burnt Mill Harlow Essex CM20 2JE
LONGMAN SOFTWARE

Tony Cross and Phil Cornes

GETTING MORE FROM THE 64

Owners of the Commodore 64 who want to use the excellent facilities of this machine without lines of relentless POKEing and PEEKing, need look no further. Part 1 of this new series unravels the mysteries of sprites.



There is a saying that a computer is only as good as its manual. Certainly, in the case of the Commodore 64, a very powerful machine is being let down by an inadequate manual. Over the next few months we hope to remedy some of the problems by describing how to get the best out of the sprite, sound and high resolution graphics facilities on this otherwise excellent machine.

In this month's article we are going to concentrate on the Commodore 64's very powerful sprite graphics facilities.

CREATING SPRITES

Sprites are user-definable high resolution graphics characters. They are displayed on a grid 24 dots wide by 21 dots high, with each dot on the grid controlling a dot on the high resolution screen. The sprites themselves are controlled by the Video Interface Chip, or VIC chip for short. To create a sprite we have to tell the VIC chip which dots to light and which dots not to light to produce the shape we require.

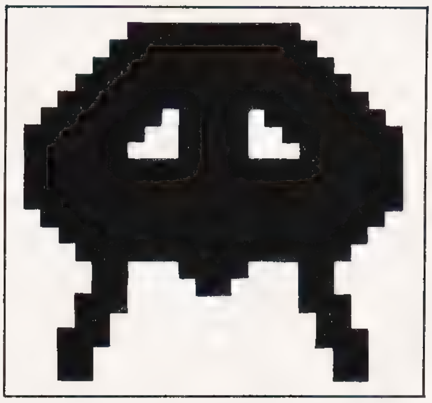
The easiest way to do this is to begin by drawing the shape of the sprite on some squared paper which is 24 squares wide by 21 squares high. Each square represents a dot on the high resolution screen. By filling in those squares which must be lit it is very easy to create the shape you require.

To enable this shape to be stored in the computer each 'filled in' square should be given the value 1 and each 'unfilled' square the value 0. By grouping the squares (or bits) together in groups of eight, we can form a 3 byte by 21 byte grid (63 bytes in all). These 63 bytes can now be stored as a contiguous 63 byte block of memory.

STORING SPRITE DATA

The problem now is to find a 'safe' place in memory to store the sprite data. The question is . . . where? Well, let's look at the problem from the VIC chip's point of view.

The VIC chip can control up to eight sprites from BASIC, each with different data. To sort out where the data for each sprite is located it looks at a group of registers called the sprite data pointers. These are at locations 2040 decimal to 2047 decimal. Each register points to the data for



one sprite. For example, 2040 points to the data for sprite 0, 2041 points to sprite 1's data and so on.

The sprite data pointers are all one byte wide: this means that they can each point to 256 different locations. As we have seen it takes 63 bytes to define a sprite. However, for a computer, the number 64 is much more convenient ($64 = 2^6$). For this reason the VIC chip assumes that each sprite's data occupies a 64-byte block of memory (the last byte is not used). When the VIC chip wants to know where a given sprite's data is located it multiplies the value in the sprite data pointer by 64 to give the address of the first byte of data. For example, if register 2040 contained 15, then the VIC chip would look at memory location 960 for the start of sprite 0's data, since 960= 15 x 64.

From this we can work out the range of memory locations which can be used to store sprite data. Since the sprite data pointers can point to 256 locations and sprites take up 64 bytes each then the range is 0 to 16320, in 64-byte blocks, as 16320= 255 x 64. (It is possible to make the sprite data pointers points to locations outside this range, but that is beyond the scope of this article).

Within the area of RAM available for sprite storage, there are many locations where it is not 'safe' to store sprite data. Figure 1 shows the general usage of the first 16K of RAM.

The first 1K of RAM is used by the Commodore 64 operating system. The only 'useable' space in here is from location 828 decimal to 1023 decimal. This is actually the cassette I/O buffer but, provided you are not using the cassette recorder within your program, you can use the buffer space to store the data for up to three different sprites. (This is where the demonstration sprites in the User Manual have been located.)

The only other available space is in the BASIC program text area. BASIC text starts at location 2048

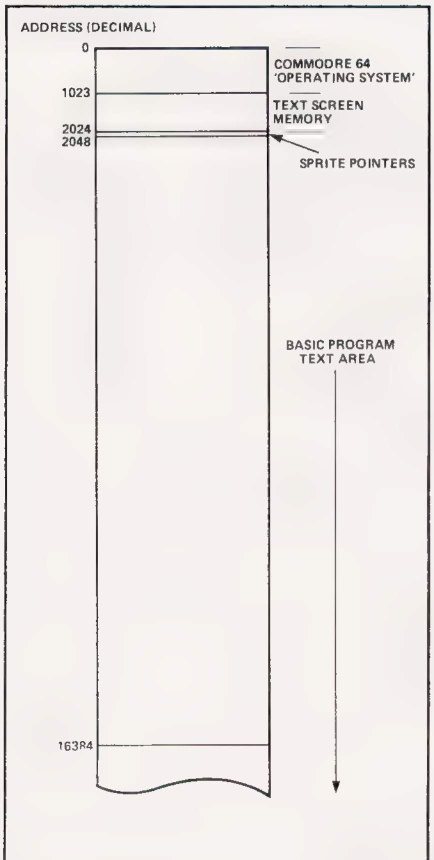


Fig. 1. Memory map for the first 16K of RAM in the 64.

decimal and extends up to 40959 decimal, giving you 38911 bytes for BASIC programs. The problem with using this area is that BASIC may overwrite your sprite data. In addition, there is one area within the BASIC text area that cannot be used for storing sprites. This is the block from 4096 decimal to 8191 decimal. The VIC chip maintains an image of the character set in this area and although BASIC text can be stored here, sprite data cannot.

It is possible to create a 'safe' storage area for sprites by moving the bottom of the BASIC text area up a few bytes. This is not easy to do from BASIC, so we have included a machine code routine at the end of this article which moves the bottom of BASIC up by 512 bytes. This creates enough room to store all eight sprites. The routine, which we have called MEMSHIFT, is called by a SYS 49329 command from BASIC (MEMSHIFT is located at address 49329). The bottom of the BASIC text area will then be moved from 2048 decimal to 2560 decimal. Pressing SHIFT and RESTORE will restore BASIC to normal. Changing the start of the BASIC text area like this will corrupt any programs which are stored in RAM, so MEMSHIFT also calls the BASIC 'NEW' routine to restore BASIC to a sensible state.

Sprites can now be safely stored in locations 2048 decimal to 2559 decimal and any BASIC programs entered from the keyboard or loaded from tape will function normally.

DISPLAYING THE SPRITES

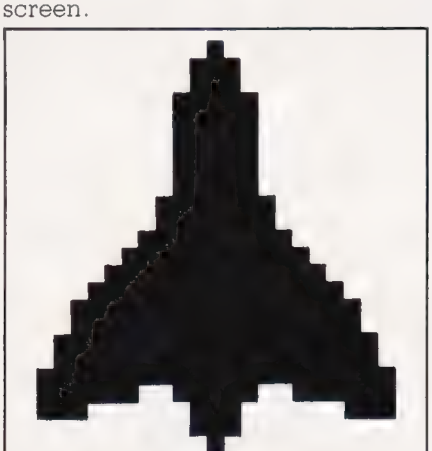
Each sprite can only be in one of two states; it is either being displayed or it is not. These two states can be represented by a 1 for 'on' and a 0 for 'off', which means that all eight sprites can be controlled by one register. This register is called the sprite enable register and it is located at 53269 decimal. Each bit within the register controls one sprite. Bit 0 controls sprite 0, bit 1 controls sprite 1 and so on.

For example, to turn sprite 3 on we need to set bit 3 of the sprite enable register to 1. Since we may not know which other sprites are already on or off, the safe way to set bit 3 is POKE 53269,PEEK (53269) OR 4.

POSITIONING SPRITES ON SCREEN

Having turned a sprite on we want to be able to position it on the screen. As we mentioned earlier, sprites are displayed on the 320 by 200 high resolution screen, so a pair of X and Y coordinates, within these ranges, is all that is required.

The 16 registers from 53248 decimal to 53263 decimal are used to store the X and Y coordinates of the eight sprites. Figure 2 shows how these registers are organised. The value held in a sprite's X and Y registers gives the position where the top left hand corner of the 24 by 21 dots grid will be placed on the high resolution screen.



However, the screen layout for sprite positioning is not numbered in the way you might expect. Instead of being numbered from 0 to 319 and from 0 to 199 the visible screen area is actually numbered from 24 to 344 in the X direction and 50 to 250 in the Y direction. The sprite screen numbering has been done in this way so that sprites can be smoothly moved on and off the screen in both the X and Y directions without requiring negative coordinates. For example, a sprite positioned at 12, 50 will be fully on screen in the Y direction and half on the screen in the X direction.

In addition, a problem occurs when you are using large values of X. This is because the X registers

SPRITE NUMBER	X REGISTER	Y REGISTER
0	53248	53249
1	53250	53251
2	53252	53253
3	53254	53255
4	53256	53257
5	53258	53259
6	53260	53261
7	53262	53263

Fig. 2.

can only hold numbers up to 255, which is not enough to completely address the 320-dots-wide screen. This has been overcome by using an additional register to provide each X register with an extra bit, thus making them all nine bits wide. (A nine-bit register can hold numbers up to 511). This additional register is called the most significant bit X register (MSBX) and it is located at 53264 decimal. Within the register bit 0 is the extra bit for sprite 0, bit 1 is the extra bit for sprite 1 and so on.

This creates a little bit of extra work for us because we now have to check every X coordinate to see if it is larger or smaller than 255. If it is larger then we must set the extra bit, subtract 256 from the X coordinate and store the result in the X register. If it is smaller, however, we must reset the extra bit and store the whole X coordinate in the X register.

DOT PAIR	DOT COLOUR
00	Transparent
01	Sprite multi-colour register 0
10	Sprite colour register
11	Sprite multi-colour register 1

Fig. 3.

To make this job a little bit easier we have included another machine code routine called SPRITEPOS which will position any of the sprites anywhere on screen. SPRITEPOS uses three BASIC integer variables, SN% , X% and Y% . SN% tells the routine which sprite to move and X% and Y% tell the routine where to put it on the screen. To use SPRITEPOS, set up the integer variables SN% , X% and Y% and then use the SYS 49217 command (SPRITEPOS is located at address 49217).

If a SYNTAX ERROR message is returned by SPRITEPOS it means that one of the three variables was missing. (Remember, they ***must*** be integer variables.) If you get an ILLEGAL QUANTITY ERROR message it means that SN% was outside the range 0-7.

USING COLOUR

The sprites can be displayed in any of the 16 available colours and in one of two 'modes'. These 'modes' are:

- The standard colour mode, where each sprite can be in one of the 16 colours on a 'transparent' grid.
- Multi-colour mode, where each sprite can be in three different colours on a 'transparent' grid.

The term 'transparent grid' means that any dots on the 24 by 21 dots sprite grid which are set to 0 will allow whatever is 'underneath' them to show through.

In standard colour mode the sprite colour is selected by POKEing the colour value (0 to 15) into the appropriate sprite colour register. These are at locations 53287 decimal to 53294 decimal. There is one register for each sprite; register 53287 is for sprite 0, register 53288 is for sprite 1 and so on, but only the low four bits are effective (four bits can take 16 different states). For example, to display sprite 5 in orange you should POKE 53292,8 (the colour value for orange is 8).

In multi-colour mode you sacrifice horizontal resolution for increased colours. Instead of treating every dot independently, as either coloured or transparent, the horizontal dots are now 'read' in pairs so that each pair of dots can be in one of three colours or transparent. Figure 3 shows how the dot pairs are coded into colours. The two sprite multi-colour registers hold the values of the two extra colours, and they are the same for all sprites in the multi-colour mode. These registers are at locations 53285 decimal for register 0 and 53286 decimal for register 1.

To select multi-colour mode for a sprite you must set the appropriate bit in the sprite multi-colour register. This is at location 53276 decimal and, as before, bit 0 is for sprite 0, bit 1 is for sprite 1 and so on. To return to standard colour mode simply reset the appropriate bit back to 0. For example, to select multi-colour mode for sprite 2 you should POKE 53276,PEEK (53276) OR 4.

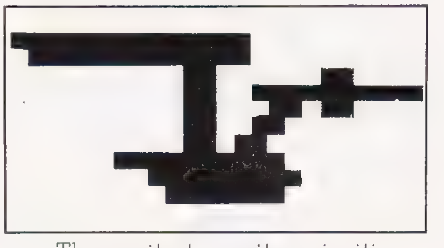
EXPANDED SPRITES

Another facility which can be used to great effect is the sprite expansion facility. Sprites can be doubled in size, in both the X and Y directions, by setting the appropriate bits in the X and/or Y expansion registers. The expansion registers are at locations 53277 for the X direction and 53271 for the Y direction. Once again, bit 0 controls sprite 0, bit 1 sprite 1 and so on in each register. For example, to display sprite 3 expanded in the X direction we must POKE 53277,PEEK (53277) OR 8.

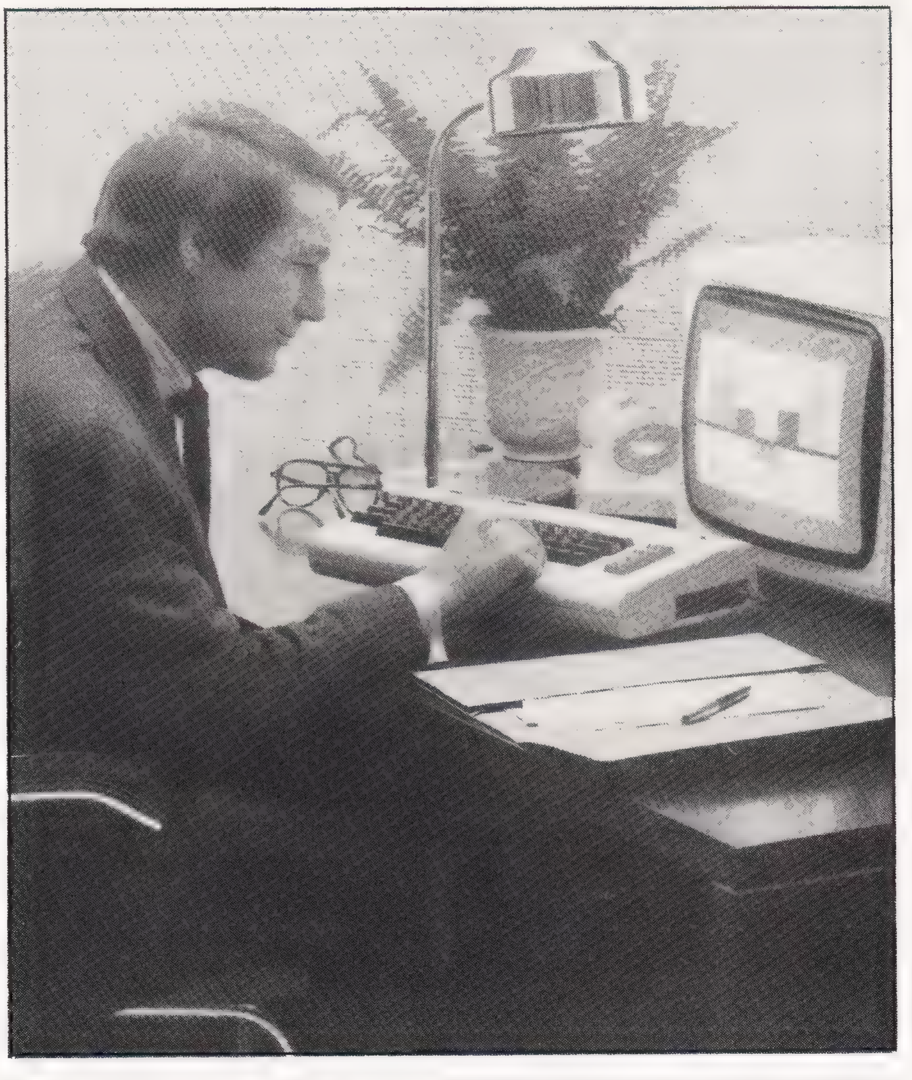
What actually happens is that with expand on, every dot on the 24 by 21 sprite grid occupies two dots on the high resolution screen instead of one. The sprite's resolution does not change; it is still 24 by 21 dots even though, with expand on in both directions, the sprite occupies 48 by 42 dots on the screen.

SPRITE PRIORITIES

One of the more powerful facilities of sprites is their ability to be assigned priorities. These priorities determine which sprite will be 'in front' when two sprites move over each other and whether a sprite will pass 'in front of' or 'behind' the background graphics.

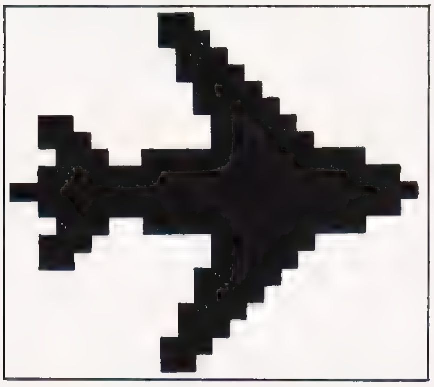


The sprite-to-sprite priorities are 'built-in' to the system. Sprite 0 has the highest priority, so no



other sprites can appear to pass in front of sprite 0. Sprite 1 has the next highest, so it will pass in front of sprits 2 to 7 but behind sprite 0. The other sprite priorities follow the same pattern.

The sprite-to-background priorities can be individually selected for each of the sprites. This is done by setting the appropriate bit in the sprite background priority register, which is located at 53275 decimal. Again, bit 0 controls sprite 0, bit 1 controls sprite 1 and so on. If the appropriate bit is a 0 then that sprite will pass in front of all



background graphics. If the bit is a 1 then that sprite will pass behind all background graphics. For example, to make sprite 4 pass behind the background graphics POKE 53275, PEEK(53275) OR 16.

DETECTING COLLISIONS

Finally we come to the most powerful facility of all, the collision detections. The VIC chip can detect when any number of sprites collide, or when any number of sprites collide with background graphics. A collision has occurred when any non-transparent areas of two sprites overlap each other, or when any non-transparent area of a sprite overlaps any background graphics.

The VIC chip reports these collisions back to us via two collision registers. These are the sprite-to-sprite collision register at location 53278 decimal, and the sprite-to-background collision register at location 53279 decimal. As before, bit 0 refers to sprite 0, bit 1 refers to sprite 1 and so on for each register.

For each of the registers the VIC chip sets the appropriate bits to 1 for those sprites which have been involved in collisions. For example, if sprites 0 and 4 have collided with each other register 53278 will contain the value 17 (bits 0 and 4 set). In both cases the collision registers retain the most recent collision information until they have been read by a PEEK. The action of PEEKing the registers automatically resets them, so if you need to retain the values in the registers you must PEEK them into variables.

THE MACHINE CODE ROUTINES

To finish for this month, here is the BASIC program to load both MEMSHIFT and SPRITEPOS into the 4K of user RAM starting at 49152 decimal. (In following articles there will be some more machine code routines which will add on to MEMSHIFT and SPRITEPOS giving you some fairly powerful graphics routines.)

These routines, and others which will be presented in later articles, require variable space in page zero. Unfortunately, Commodore have used nearly all of page zero locations for various system functions. So in order to create space for the variables in our routines we have 'borrowed' the locations used by the KERNAL RS-232 routines. This means that if you are using the RS-232 port you won't be able to use our routines.. Sorry about that....!

```
10 REM Routine to load MEMSHIFT and SPRITEPOS
20 FOR A=0 TO 197
30 READ Q : POKE 49152+A,Q
40 NEXT A
100 DATA 165,46,133,254,165,45,133,253
110 DATA 160,0,165,254,197,48,208,8
120 DATA 165,253,197,47,208,2,56,96
130 DATA 177,253,197,253,208,17,200,177
140 DATA 253,197,251,208,11,200,177,253
150 DATA 170,200,177,253,168,24,96,200
160 DATA 152,24,105,6,161,253,153,253
170 DATA 165,254,105,0,133,254,76,8
180 DATA 192,169,211,133,252,169,206,133
190 DATA 251,32,0,192,176,89,152,48
200 DATA 91,201,8,176,87,137,249,10
210 DATA 137,250,169,1,206,136,240,4
220 DATA 10,76,93,192,133,248,169,217
230 DATA 133,[illegible],169,[illegible],133,251,32,0
240 DATA 192,176,[illegible],166,[illegible],[illegible],[illegible],[illegible]
250 DATA 208,169,[illegible],133,[illegible],[illegible],[illegible],[illegible]
260 DATA [illegible],32,0,192,176,[illegible],[illegible],[illegible]
270 DATA 166,[illegible],152,[illegible],[illegible],[illegible],[illegible],[illegible]
280 DATA [illegible],[illegible],[illegible],165,[illegible],[illegible],[illegible],[illegible]
290 DATA [illegible],[illegible],[illegible],[illegible],[illegible],[illegible],[illegible],[illegible]
300 DATA [illegible],[illegible],208,141,[illegible],[illegible],[illegible],[illegible]
310 DATA [illegible],[illegible],[illegible],[illegible],[illegible],[illegible],[illegible],[illegible]
320 DATA [illegible],[illegible],[illegible],[illegible],[illegible],[illegible],[illegible],[illegible]
330 DATA [illegible],[illegible],[illegible],[illegible],[illegible],[illegible],[illegible],[illegible]
340 DATA 141,0,[illegible],[illegible],[illegible],[illegible]
```

Listing 1. BASIC loader for the MEMSHIFT and SPRITEPOS routines.

ELIMINATE FAULTY CASSETTES

DataClone is the first company in the UK established specifically for the duplication of data cassettes.

All other duplicating houses are audio oriented - only DataClone has a duplicating system designed from scratch purely to handle computer information.

The result?

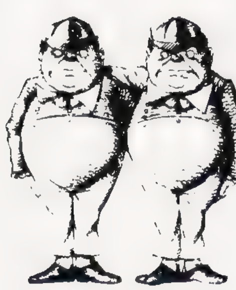
Greatly improved reliability in data transfer rates from 300 to beyond 2400 baud - previously unattainable.

All formats catered for. Quantities from 200 to infinity.

Contact us now for brochure.

Dataclone - the first specialist service for computer cassettes.

DATACLONE HAS EXPANDED, PRODUCING AN EVEN FASTER TURNAROUND. NOTE NEW ADDRESS



DATACLONE

UNIT 1,
ROSLIN SQUARE
ROSLIN ROAD
ACTON, LONDON W3
TEL: 01-993 2134
TELEX: 21879

Which home computer gives you so much software for so little?

Entertainment for all:

Video Chess	£34.95
Munchman	£29.95
Parsec (Optional Speech)	£29.95
Othello	£24.95
Tunnels of Doom	£24.95
Adventure Cartridge	£24.95

Additional Adventure Games: all at £14.95

Adventure Land
Mission Impossible
Voodoo Castle
The Count
Strange Odyssey
Mystery Fun House
Pyramid of Doom
Ghost Town
Savage Island
Golden Voyage

Alpiner (Optional Speech)	£24.95
TI Invaders	£19.95
Car Wars	£19.95
Chisholm Trail	£19.95
Hustle	£14.95
Tombstone City	£14.95
Connect Four	£14.95
Video Games 1	£14.95
Hunt The Wumpus	£14.95
Five-A-Side Soccer	£14.95
Amazing	£14.95
Attack	£14.95
Blasto	£11.95
Blackjack & Poker	£11.95
Zero Zap	£11.95
Yahtzee	£11.95
Oldies But Goodies 1	£11.95
Oldies But Goodies 2	£11.95
Market Simulation	£11.95

Educational:

Early Reading	£29.95
Music Maker	£29.95
Alligator Mix	£24.95
Alien Addition	£24.95
Demolition Division	£24.95
Dragon Mix	£24.95
Minus Mission	£24.95
Meteor Multiplication	£24.95
Touch Typing Tutor	£24.95
Addition Subtraction 1	£19.95
Addition Subtraction 2	£19.95
Multiplication	£19.95
Division	£19.95
Numeration 1	£19.95
Numeration 2	£19.95
Early Learning Fun	£14.95
Beginning Grammar	£14.95
Number Magic	£14.95
Hangman	£11.95
Teach Yourself Extended BASIC	£11.95
Beginners BASIC Tutor	£9.95

Home Organisation:

Personal Record Keeping	£39.95
Personal Report Generator	£39.95
Home Budget	£24.95
Home Financial Decisions	£24.95
Personal Financial Aids	£11.95

Other Applications:

Inventory	£85.95
Invoicing	£85.95
Mailing List	£85.95
TI Writer (Word Processing)	£85.95
Multiplan (Spread Sheet Program)	£85.95
Statistics	£54.95
Terminal Emulator	£49.95
Maths Routine Library	£29.95
Electrical Engineering Library	£29.95
Graphing Package	£29.95
Structural Engineering Library	£29.95
Programming Aids 2	£19.95
Programming Aids 3	£19.95
Speech Editor	£19.95
Programming Aids 1	£11.95

Programming Languages:

PASCAL Editor	£99.95
PASCAL Linker	£79.95
Extended BASIC	£69.95
TI Logo	£69.95
Editor/Assembler	£69.95
Mini Memory	£69.95
PASCAL Compiler	£59.95

A.S.K. Applied Systems Knowledge:

Hide and Seek	t.b.a.
Number Gulper	t.b.a.

Collins-Educational:

TI-99/4A Starter Pack 1	£9.95
TI-99/4A Starter Pack 2	£9.95
TI-99/4A Game Writer Pack 1	£9.95
TI-99/4A Game Writer Pack 2	£9.95
Chess Learner Pack	£9.95
Record Keeper Pack	£9.95

Ivan Berg Software:

Maths Tester 1	£9.95
Maths Tester 2	£9.95
Physics Tester	£9.95
Chemistry Tester	£9.95
Biology Tester	£9.95
Human Biology Tester	£9.95

Bond Associates:

Easycalc	£49.95

Little Genius Ltd:

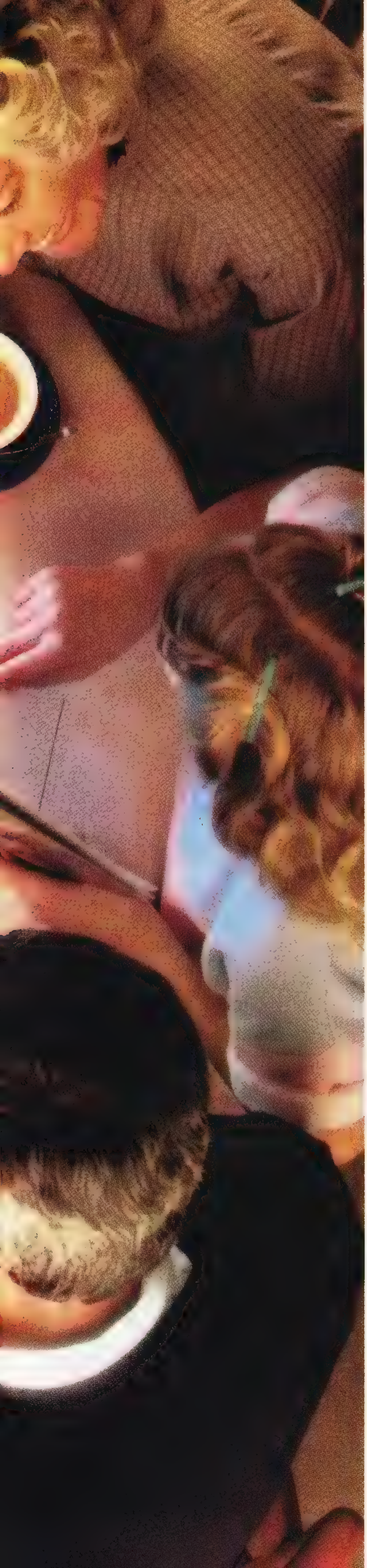
Scrabble	£29.95

And is available at all these dealers?

All branches of:
Argos, Comet, Dixons, Greens of Debenhams, John Lewis, Rumbelows, Wigfalls, Zappo.
Major branches of:
Asda, Binns, Computers for All, CO-OP, Currys, Fine Fare, Ketts, Photomarket, Rymans, Spectrum, Telefusion.
And at:
ABC Computers – St Austell
Akhter – Harlow
Anglia Audio – Bedford
Anglia Sound – Stevenage
Audio Marketing – London
Audio Vision – Faversham
Bagnall – Stafford
Carvells – Rugby
Combined Trading – Hatfield
Computer Supermarket – Manchester
Cotton TV – Peterborough
Dean and Son – London
Delta Electronics – London
Densham Computers – Poole
Desk Aids – Southampton
Dodar – Ashton
Fenwicks – Newcastle
Galaxy Video – Maidstone
Hamleys – London
Harrods – London
Heffers – Cambridge
Hyman Computers – Manchester
Landau – Sutton
Lion House – London
Micro Value – Amersham, Bucks.
Midshires – Crewe
Milequip – Gloucester
OEM Computers – Rugby
Parco Electronics – Honiton
REW – London
Robox – Glasgow
Science Studio – Oxford
Selfridges – London
Star Trek Video – Wigan
Toy and Hobby – Wigan
Universal Warehouse – Reading
Video Palace – London
Vision Store – Kingston
Welwyn Dept. Store – Welwyn
And many other leading Computer Stores.

Turn the page and see the unbeatable.


NEW LOW PRICE
£109.95
OR LESS
TEXAS INSTRUMENTS HOME COMPUTER



The unbeatable TI Home Computer. It's all the computers your family will ever need.

Buying a home computer is something you have to get right first time. It's too late when you've got one to find it won't take plug-in software. Or can't be programmed without an expensive accessory.

The TI Home Computer is a real computer system

The TI Home Computer has got the memory power you might expect from more expensive computers, built in. At its heart is a powerful TMS 9900 16-BIT Microprocessor. Most other home computers have only an 8-BIT. And you can expand the memory from 16K of RAM up to 52K.

The total memory capacity is 114K Bytes.

A wide range of software for everyone

Another feature that makes the TI system so powerful, yet so easy to use is Solid State Software.™ These plug-in cartridges cover everything from space games like Parsec™ to teaching maths, managing home finances and composing music. And the range is getting wider all the time.

It even has what professionals look for in a home computer

CPU: TMS 9900 16-BIT, plus 256-byte Scratchpad RAM.
Memory: Total 114K bytes; 26K bytes ROM internal; up to 36K ROM cartridges external; 16K built-in RAM expandable to 52K bytes.
Keyboard: 48 Key QWERTY, alphalock, function key auto repeat.
Sound: 5 octaves, 3 simultaneous tones, noise tone.
Colour: 16 foreground and background. High resolution.
Interfaces: Cassette, TV, 2 joysticks, main peripheral port.

™ trademark of Texas Instruments.

More than one programming language

The standard programming language, TI BASIC, is built into your TI Home Computer so you can begin programming right away. But there's an expanded range of optional languages like Extended BASIC, TI Logo, USCD-Pascal, TI FORTH and Assembler.

With these you can fully expand your programming skills.

A wide range of peripherals

Most computers lose a lot of memory when you add peripherals. The TI Home Computer is different. Every peripheral comes with its own built-in programs to keep the loss of memory to a minimum.

The convenient Peripheral Expansion System houses up to eight peripherals. Additional hardware cards simply plug in. You can even add a complete Floppy Disk Memory System.

The peripherals include memory expansion, RS232 Interface, P-Code card and more. There's also a sophisticated matrix printer and Solid State Speech™ synthesizer – which you can use with your own TI BASIC programs.

A lot more for no more

The TI Home Computer gives you so much more without costing more. At today's price it's exceptional value. Take your family round to try one. If you never try it you'll never know what you're missing.

TEXAS INSTRUMENTS

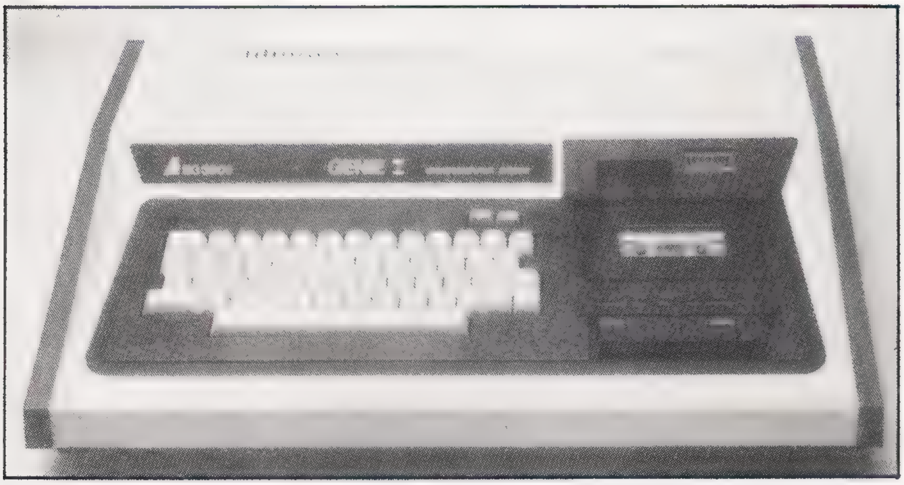
Creating useful products and services for you.


ASP SOFTWARE
A STEP BEYOND THE ARCADE...
Demon Knight · The Valley · Strategy 1 – Invasion · Strategy 2 – Bismark
White Barrows · Detective · Cells & Serpents · Stockmarket
Conquering Everest · Cloneit/Renumber · Gallery of Death · Planetfall
HORNSBY

Andrew Howard

MCBAS CONVERTER

Storing machine-code programs in BASIC programs as DATA statements and POKEing the values into memory where required is a standard technique. This utility makes the machine-code-to-BASIC conversion simple and error-free.



MCBAS is a short and fast-executing program that will take any machine code program in memory and convert each byte of it into decimal, placing the result in a DATA statement. Thus BASIC programs can simply POKE the machine code routine into memory rather than load it separately. The program uses a mere 763 bytes of user memory and can convert 12288 bytes of machine code into BASIC in as little as 30 seconds.

Three new commands, OPEN, CLOSE and GET provide full control of the program.

The program was written for a 16K Level II Genie and occupies memory locations 4300 hex to 45FB hex: therefore it cannot be used with Level III or DOS. If the listing is typed in as printed, upon loading with the SYSTEM command a message will be displayed at the top of the screen and when complete the program will automatically execute. No memory reservation is required and BASIC programs will automatically reside above MCBAS.

OPEN AND CLOSE

The CLOSE command will 'shut off' the BASIC program resident in memory so that further inputs from the keyboard or from tape may be carried out. When OPEN is executed the second program will be appended to the first. Obviously the two program's line numbers must not conflict and the second program's line numbers must be higher than the first. OPEN initially asks the question "OPEN?" and the Y key must be pressed to make the open. Also, the command does not check for bad line numbers between the two programs, so it is up to you.

If several programs have been CLOSEd, OPEN will open all of them.

THE GET COMMAND

The GET command gets the machine code into BASIC DATA statements. The format of this command is:

```
GET,start address,end address,
start line,increment(,LIST)(Newline)
```

The 'LIST is optional, and if supplied the program will be listed as it is formed. For example:

```
GET,28672,32767,10,2  (Newline)
```

will convert the program from 28672 to 32767 and put it into DATA statements starting at line 10 with an increment of 2. The BASIC program will not be listed as it is formed.

Upon Newline your parameters will be printed on the screen and you will be asked to press a key. Break will abort the conversion, any other key will continue it. Upon completion the READY message will be printed. During conversion the Break key will intervene.

Each of the numeric parameters (start, end, start line and increment) should all be expressed as integers in the normal way, that is if any of them are greater than 32767 they should be expressed as negative numbers. For example, to convert a program from F000 (hex) or 61440 (decimal) to FFFF (hex) or 65535 (decimal) you should type GET, −4096, −1, start line..., since −4096 = 61440 — 65536 and −1 = 65535 — 65536. The start line and increment values should also be expressed in this way.

Each line in a converted machine code program is, on average, about 192 characters in length.

HOW IT WORKS

The first two lines of program set the jump so that instead of the SYSTEM command displaying a second *? control is automatically transferred to the initialistion program. Lines 30 and 40 are the loading message. The program begins at 4300 hex and lines 60 to 430 are the program's messages. Lines 440 through 490 are all the program's variables.

The initialisation routine INIT first disables the SYSTEM auto-execute by replacing the JP instruction in 41E2 hex with a RETurn and sets the pointer to the new start of BASIC programs. The NEW routine in ROM (at 1B4A hex) is then called and the title message printed. Lines 570 to 660 set all of the new ROM exits for the OPEN, CLOSE and GET commands and finally control is returned to the READY message with a JP 0072 hex instruction.

The CLOSE routine (at line 680) first prints the 'Program Closed Off' message. The pointer to the start of the simple variables (non-array variables) is decremented twice and stored as the new pointer to the start of BASIC programs so that any more program lines will be stored directly above the old program. The NEW routine is then executed and control returned to the READY message.

The OPEN routine (line 760) prints the 'Open ?' message and calls the ROM routine at 0049 hex which loops round, only returning when a key is pressed (the ASCII code for the key will be in the accumulator). If the Y key is not pressed then control is returned to the READY message. Otherwise the 'Program Opened' message is printed and the pointer to the start of BASIC programs is reset to just after the end of MCBAS, as done by the INIT routine. This will 'open up' all BASIC programs in memory.

On entry, the GETLIN subroutine expects the HL register pair to point to the next character in the input stream. The routine then checks for a comma (Syntax Error if not) and converts the expression following it to integer. On exit, HL will point to the first character following the number and DE will contain the integer result of the expression.

The GET routine has a fairly long initialising section (lines 890 to 1440). First, the sub-title 'Machine Code to Basic' is printed. At this point, HL points to the first non-blank character following the GET. The start address parameter is evaluated by calling the GETLIN routine. On return DE will contain the result and HL will point to the start of the next parameter. The contents of DE are stored in the START variable and the remaining parameters (end, start line and increment) are evaluated, the results being stored in the appropriate variable. Next a test for a comma is made and if true the following character is tested for the LIST token (syntax error if not LIST). A variable LIST is set to one to indicate that the program is to be listed as converted. If there was no comma then this variable is zeroed.

The next section of initialisation displays each of the parameters. The four numerics are displayed by first printing a message saying what the number is and then the number itself. Then HL is loaded with the ASCII equivalents of two 'F's', or an 'N' and a space, according to whether the LIST is zero or one respectively. HL is then stored at the address following the '0' in 'List Switch...' and the whole message printed. Lines 1360 to 1400 print the 'Press any key' prompt and control is returned to the STOP routine if the Break key is pressed. Otherwise the 'Assembling...' message is printed and the flag ENDFG is zeroed.

BUILDING THE DATA

The NEWLIN routine initialises a new program line by converting the current line number in the LINE variable to ASCII and storing it in the input buffer. A space and the word 'DATA' is then stored and the current position in the input buffer is transferred to DE. B is loaded with the number of characters in the input buffer already plus two (12) and HL is loaded with the current address in the machine code program (stored in START) and the line build routine is executed.

The BUILD routine loads the A register with the current byte of the machine code program and call the NUMBER subroutine. This subroutine converts the number in A to ASCII and stores the string in a buffer at 405E hex. All leading zeroes are then removed and the remaining portion of the number being transferred to the input buffer (with DE pointing to the next empty position). If the number was all zeroes then a single zero is stored in the input buffer and DE incremented to the next position. In either case, control is then returned to the BUILD routine.

Next, the current machine code program address (in START) is compared with the end address. On equality the ENDFG flag is set to one and control transferred to the LINFUL routine at line 1910. Otherwise the B register is incremented and if it then contains 69 then that particular program line is full and control is transferred to LINFUL. Otherwise a comma is stored in the input buffer, DE incremented to the next available position and HL incremented to the next byte in the machine code programe. Control is transferred to the start of the BUILD routine to get the next byte.

The LINFUL routine increments HL (pointer into the machine code program) and stores it in START as the current pointer. A zero is stored in the input buffer to mark the end and the LIST flag is tested. If it is 1 then the input buffer is printed together with a carriage return.

The current line number (in LINE) is added to the increment (in INCREM) and the result is stored in LINE as the next current line number. Then a special jump is set at a ROM exit at 41B8 hex so that control is returned to MCBAS. HL is loaded with the address of the input buffer minus one and the ROM routine at 1A81 hex is executed. This will take the contents of the input buffer, tokenize it and merge it into the BASIC program. Control will then be returned to CONTMC (line 2140) which disables the jump (by replacing it with RETurn) and tests the ENDFG variable. If it is 1 then the conversion is complete and control is returned to READY (by JP 0072 hex). Otherwise the keyboard is tested. If Break is pressed then the STOP routine is executed. If Shift-@ is pressed then the program waits for another key depression before continuing. Otherwise control is returned to the NEWLIN routine to form another program line.

Listing 1. The MCBAS assembler listing: for brevity the text messages are given as a hex dump.

```
3CØØ    4C 4F 41 44 49 4E 47 2Ø 4D 43 42 41 53 2Ø 43 4F
3C1Ø    4E 56 45 52 54 45 52 2Ø 2Ø 2Ø 2Ø 2Ø

43ØØ    4D 43 42 41 53 2Ø 56 49 44 45 4F 2Ø 47 45 4E 49
431Ø    45 2Ø 4D 41 43 48 49 4E 45 2Ø 43 4F 44 45 2Ø 54
432Ø    4F 2Ø 42 41 53 49 43 2Ø 43 4F 4E 56 45 52 54 45
433Ø    52 ØD 42 59 2Ø 41 4E 44 52 45 57 2Ø 48 4F 57 41
434Ø    52 44 2C 2Ø 31 33 54 48 2Ø 4D 41 52 43 48 2C 2Ø
435Ø    31 39 38 33 2E ØD 43 4F 5Ø 59 52 49 47 48 54 2Ø
436Ø    31 39 38 33 2E ØD ØØ 4D 41 43 48 49 4E 45 2Ø 43
437Ø    4F 44 45 2Ø 54 4F 2Ø 42 41 53 49 43 ØD ØØ ØD 5Ø
438Ø    52 4F 47 52 41 4D 2Ø 49 53 2Ø 4F 5Ø 45 4E 45 44
439Ø    ØD ØØ 5Ø 52 4F 47 52 41 4D 2Ø 49 53 2Ø 43 4C 4F
43AØ    53 45 44 2Ø 4F 46 46 ØD ØØ 4F 5Ø 45 4E 2Ø 3F 2Ø
43BØ    ØE ØØ 53 54 41 52 54 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø 3A 2Ø
43CØ    ØØ ØD 45 4E 44 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø 3A 2Ø
43DØ    ØØ ØD 53 54 41 52 54 2Ø 4C 49 4E 45 2Ø 2Ø 3A 2Ø
43EØ    ØØ ØD 49 4E 43 52 45 4D 45 4E 54 2Ø 2Ø 2Ø 3A 2Ø
43FØ    ØØ ØD 4C 49 53 54 2Ø 53 57 49 54 43 48 2Ø 3A 2Ø
44ØØ    4F 46 46 ØØ ØA ØA 5Ø 52 45 53 53 2Ø 41 4E 59 2Ø
441Ø    4B 45 59 ØA ØE ØØ ØD 41 53 53 45 4D 42 4C 49 4E
442Ø    47 2Ø 42 41 53 49 43 2Ø 5Ø 52 4F 47 52 41 4D

41E2    C3 3D 44        ØØØ2Ø  AUTO    JP      INIT

442F    ØA ØF           ØØ42Ø          DEFW    ØFØAH
4431    ØD ØØ           ØØ43Ø          DEFW    13
4433    ØØ              ØØ44Ø  START   DEFS    2

4435                    ØØ45Ø  END     DEFS    2
4437    ØØ              ØØ46Ø  LIST    NOP
4438                    ØØ47Ø  LINE    DEFS    2
443A                    ØØ48Ø  INCREM  DEFS    2
443C    ØØ              ØØ49Ø  ENDFG   NOP
443D    3E C9           ØØ5ØØ  INIT    LD      A,2Ø1
443F    32 E2 41        ØØ51Ø          LD      (41E2H),A
4442    21 FA 45        ØØ52Ø          LD      HL,BASIC
4445    22 A4 4Ø        ØØ53Ø          LD      (4ØA4H),HL
4448    CD 4A 1B        ØØ54Ø          CALL    1B4AH
444B    21 ØØ 43        ØØ55Ø          LD      HL,TITLE
444E    CD 75 2B        ØØ56Ø          CALL    2B75H
4451    3E C3           ØØ57Ø          LD      A,195
4453    32 79 41        ØØ58Ø          LD      (4179H),A
4456    32 85 41        ØØ59Ø          LD      (4185H),A
4459    32 7F 41        ØØ6ØØ          LD      (417FH),A
445C    21 85 44        ØØ61Ø          LD      HL,OPEN
445F    22 7A 41        ØØ62Ø          LD      (417AH),HL
4462    21 71 44        ØØ63Ø          LD      HL,CLOSE
4465    22 86 41        ØØ64Ø          LD      (4186H),HL
4468    21 A7 44        ØØ65Ø          LD      HL,GET
446B    22 8Ø 41        ØØ66Ø          LD      (418ØH),HL
446E    C3 72 ØØ        ØØ67Ø          JP      72H
4471    21 92 43        ØØ68Ø          LD      HL,MSCLSE
4474    CD 75 2B        ØØ69Ø          CALL    2B75H
4477    2A F9 4Ø        ØØ7ØØ          LD      HL,(4ØF9H)
447A    2B              ØØ71Ø          DEC     HL
447B    2B              ØØ72Ø          DEC     HL
```

```
447C  22 A4 40           LD    (4ØA4H),HL
447F  CD 4D 1B           CALL  1B4DH
4482  C3 72 ØØ           JP    72H
4485  21 A9 43     OPEN  LD    HL,QUOPEN
4488  CD 75 2B           CALL  2B75H
448B  CD 49 ØØ           CALL  49H
448E  FE 59              CP    'Y'
449Ø  C2 72 ØØ           JP    NZ,72H
4493  21 7E 43           LD    HL,MSOPEN
4496  CD 75 2B           CALL  2B75H
4499  21 FA 45           LD    HL,BASIC
449C  22 A4 4Ø           LD    (4ØA4H),HL
449F  C3 72 ØØ           JP    72H
44A2  CF           GETLIN RST  8
44A3  2C                 DEFM  ','
44A4  C3 Ø2 2B           JP    2BØ2H
44A7  E5           GET   PUSH  HL
44A8  21 67 43           LD    HL,SUBTTL
44AB  CD 75 2B           CALL  2B75H
44AE  E1                 POP   HL
44AF  CD A2 44           CALL  GETLIN
44B2  ED 53 33 44        LD    (START),DE
44B6  CD A2 44           CALL  GETLIN
44B9  ED 53 35 44        LD    (END),DE
44BD  CD A2 44           CALL  GETLIN
44CØ  ED 55 38 44        LD    (LINE),DE
44C4  CD A2 44           CALL  GETLIN
44C7  ED 53 3A 44        LD    (INCREM),DE
44CB  2B                 DEC   HL
44CC  D7                 RST   1ØH
44CD  FE 2C              CP    ','
44CF  2Ø ØA              JR    NZ,NOLIST
44D1  D7                 RST   1ØH
44D2  FE B4              CP    ØB4H
44D4  C2 97 19           JP    NZ,1997H
44D7  3E Ø1              LD    A,1
44D9  18 Ø1              JR    SETLST
44DB  AF           NOLIST XOR  A
44DC  32 37 44     SETLST LD   (LIST),A
44DF  21 B2 43           LD    HL,MSSTRT
44E2  CD 75 2B           CALL  2B75H
44E5  2A 33 44           LD    HL,(START)
44E8  CD AF ØF           CALL  ØFAFH
44EB  21 C1 43           LD    HL,MSEND
44EE  CD 75 2B           CALL  2B75H
44F1  2A 35 44           LD    HL,(END)
44F4  CD AF ØF           CALL  ØFAFH
44F7  21 D1 43           LD    HL,MSLINE
44FA  CD 75 2B           CALL  2B75H
44FD  2A 38 44           LD    HL,(LINE)
45ØØ  CD AF ØF           CALL  ØFAFH
45Ø3  21 E1 43           LD    HL,MSINC
45Ø6  CD 75 2B           CALL  2B75H
45Ø9  2A 3A 44           LD    HL,(INCREM)
45ØC  CD AF ØF           CALL  ØFAFH
45ØF  21 46 46           LD    HL,4646H
4512  3A 37 44           LD    A,(LIST)
4515  B7                 OR    A
4516  28 Ø3              JR    Z,NOTLST
4518  21 4E 2Ø           LD    HL,2Ø4EH
451B  22 Ø1 44     NOTLST LD   (MSLONF+1),HL
451E  21 F1 43           LD    HL,MSLIST
4521  CD 75 2B           CALL  2B75H
4524  21 Ø4 44           LD    HL,ANYKEY
4527  CD 75 2B           CALL  2B75H
452A  CD 49 ØØ           CALL  49H
452D  3D                 DEC   A
452E  CA AA 1D           JP    Z,1DAAH
4531  21 16 44           LD    HL,ASSEMB
4534  CD 75 2B           CALL  2B75H
4537  AF                 XOR   A
4538  32 3C 44           LD    (ENDFG),A
453B  2A 38 44     NEWLIN LD   HL,(LINE)
453E  CD 9A ØA           CALL  ØA9AH
4541  Ø1 ØØ ØØ           LD    BC,Ø
4544  2A A7 4Ø           LD    HL,(4ØA7H)
4547  CD 2F 13           CALL  132FH
454A  36 2Ø              LD    (HL),32
454C  23                 INC   HL
454D  36 44              LD    (HL),'D'
454F  23                 INC   HL
455Ø  36 41              LD    (HL),'A'
4552  23                 INC   HL
4553  36 54              LD    (HL),'T'
4555  23                 INC   HL
4556  36 41              LD    (HL),'A'
4558  23                 INC   HL
4559  EB                 EX    DE,HL
455A  Ø6 ØC              LD    B,12
455C  2A 33 44           LD    HL,(START)
455F  E5           BUILD PUSH  HL
456Ø  7E                 LD    A,(HL)
4561  6F                 LD    L,A
4562  26 ØØ              LD    H,Ø
4564  CD 9A ØA           CALL  ØA9AH
4567  C5                 PUSH  BC
4568  Ø1 ØØ ØØ           LD    BC,Ø
456B  CD D9 45           CALL  NUMBER
456E  C1                 POP   BC
456F  E1                 POP   HL
457Ø  D5                 PUSH  DE
4571  ED 5B 35 44        LD    DE,(END)
4575  DF                 RST   18H
4576  D1                 POP   DE
4577  2Ø Ø7              JR    NZ,OK
4579  3E Ø1              LD    A,1
457B  32 3C 44           LD    (ENDFG),A
457E  18 ØE              JR    LINFUL
458Ø  Ø4           OK    INC   B
4581  3E 45              LD    A,45H
4583  B8                 CP    B
4584  28 Ø8              JR    Z,LINFUL
4586  EB                 EX    DE,HL
4587  36 2C              LD    (HL),2CH
4589  23                 INC   HL
458A  EB                 EX    DE,HL
458B  23                 INC   HL
458C  18 D1              JR    BUILD
458E  23           LINFUL INC  HL
458F  22 33 44           LD    (START),HL
4592  EB                 EX    DE,HL
4593  36 ØØ              LD    (HL),Ø
4595  3A 37 44           LD    A,(LIST)
4598  B7                 OR    A
4599  28 ØB              JR    Z,SKPLST
459B  2A A7 4Ø           LD    HL,(4ØA7H)
459E  CD 75 2B           CALL  2B75H
45A1  3E ØD              LD    A,13
45A3  CD 33 ØØ           CALL  33H
45A6  2A 3A 44     SKPLST LD   HL,(INCREM)
45A9  EB                 EX    DE,HL
45AA  2A 38 44           LD    HL,(LINE)
45AD  19                 ADD   HL,DE
45AE  22 38 44           LD    (LINE),HL
45B1  3E C3              LD    A,195
45B3  21 C3 45           LD    HL,CONTMC
45B6  32 B8 41           LD    (41B8H),A
45B9  22 B9 41           LD    (41B9H),HL
45BC  2A A7 4Ø           LD    HL,(4ØA7H)
45BF  2B                 DEC   HL
45CØ  C3 81 1A           JP    1A81H
45C3  3E C9        CONTMC LD   A,2Ø1
45C5  32 B8 41           LD    (41B8H),A
45C8  3A 3C 44           LD    A,(ENDFG)
45CB  B7                 OR    A
45CC  C2 72 ØØ           JP    NZ,72H
45CF  CD 58 Ø3           CALL  358H
45D2  B7                 OR    A
45D3  C4 AØ 1D           CALL  NZ,1DAØH
45D6  C3 3B 45           JP    NEWLIN
45D9  D5           NUMBER PUSH DE
45DA  21 5E 4Ø           LD    HL,4Ø5EH
45DD  E5                 PUSH  HL
45DE  CD 2F 13           CALL  132FH
45E1  E1                 POP   HL
45E2  Ø6 Ø5              LD    B,5
45E4  7E           NLOOP LD    A,(HL)
45E5  FE 3Ø              CP    3ØH
45E7  2Ø Ø9              JR    NZ,NONZER
45E9  23                 INC   HL
45EA  1Ø F8              DJNZ  NLOOP
45EC  D1                 POP   DE
45ED  3E 3Ø              LD    A,3ØH
45EF  12                 LD    (DE),A
45FØ  13                 INC   DE
45F1  C9                 RET
45F2  D1           NONZER POP  DE
45F3  48                 LD    C,B
45F4  Ø6 ØØ              LD    B,Ø
45F6  ED BØ              LDIR
45F8  C9                 RET
45F9  ØØ                 NOP
45FA  ØØ ØØ        BASIC DEFW  Ø
443D                     END   INIT
```

ANYKEY	4404	CONTMC	45C3	INIT	443D	MSINC	43E1	NEWLIN	453B	OK	458Ø
ASSEMB	4416	END	4435	LINE	4438	MSLINE	43D1	NLOOP	45E4	OPEN	4485
AUTO	41E2	ENDFG	443C	LINFUL	458E	MSLIST	43F1	NOLIST	44DB	QUOPEN	43A9
BASIC	45FA	GET	44A7	LIST	4437	MSLONF	44ØØ	NONZER	45F2	SETLST	44DC
BUILD	455F	GETLIN	44A2	MSCLSE	4392	MSOPEN	437E	NOTLST	451B	SKPLST	45A6
CLOSE	4471	INCREM	443A	MSEND	43C1	MSSTRT	43B2	NUMBER	45D9	START	4433
SUBTTL	4367	TITLE	43ØØ								

Some new hoops for the Dragon to jump through.

Unless you can already write software programs, the usefulness of a home computer is governed, to a great extent, not by its power but by the software available.

After all what's the use of a very large memory if you can't use it properly?

So at Dragon we not only took great care when designing the actual computer, we also made sure that the people who designed the software took equal care.

By using only top writers, and by giving them subjects of interest to you, we managed to produce some of the very best programs around.

Take, for example, the Cyrus Chess program. Designed by a grand master it, not surprisingly, won the accolade of Microcomputer Chess Champion.

Of course you don't have to be a grand master to play it, but, on level nine, you do need to be one to beat it.

Mind you, that's just one of nearly a hundred Dragon software titles available and, though they're all fun, they're certainly not all games.

There are programs that can teach your children spelling and mathematics.

Another can teach you how to type.

And, although some of our other titles are displayed above, the best way of seeing our full range of software isn't by looking at this ad.

It's by going down to your local Dragon stockist.

DRAGON

The problem with buying a home computer, as you may already have discovered, is there's often very little software to go with it. Or all that is available is games, games and more games.

There's no such problem, however, with the Commodore 64. It has a more extensive range of serious software than any other home computer.

It also has an unusually large (in fact elephantine) 64K memory, as well as every peripheral you're ever likely to need.

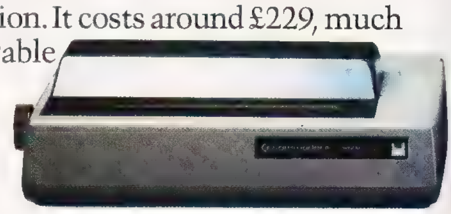
Put simply, this means the computer has the capacity to run more interesting, entertaining and complex programs.

You can teach yourself just about any subject you care to mention, even computer programming.

And for the office there are programs like word processing, financial planning, information storage and stock control.

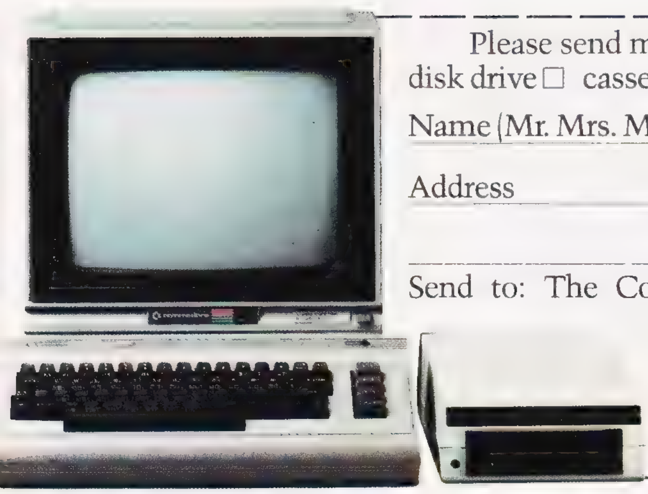
Finally, when you're mentally exhausted, you can even entertain yourself – yes, with games.

When all's said and done, however, we do have to admit that in one respect the Commodore 64 isn't up with the competition. It costs around £229, much less than any comparable machine.



And that's a fact we hope you'll never ever forget.

When you have an enormous memory there's no end to the things you can do.



Please send me further information on: the 64 computer ☐ 64 software ☐ disk drive ☐ cassette unit ☐ printers ☐ monitor ☐

Name (Mr. Mrs. Miss) ______________________

Address ______________________

____

Send to: The Commodore Information Centre, 675 Ajax Avenue, Slough, Berkshire SL1 4BG. Or telephone (0753) 79292.

commodore

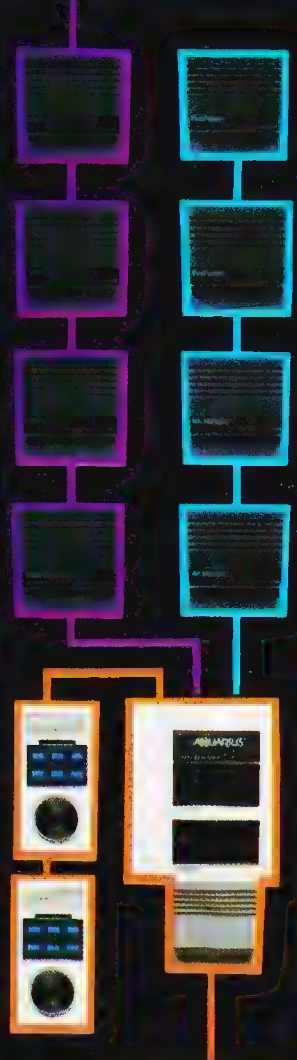
6S CTO 1283

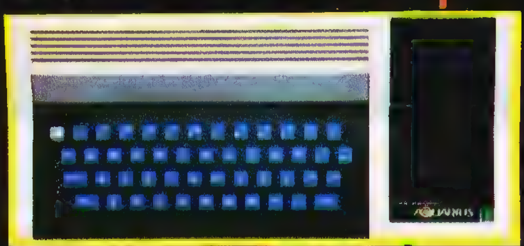
This is an entirely new computer system. It was designed with both eyes fixed firmly on the future. So that whatever shape the future takes, we'll be able to fit it into the system. Just as simply as the peripherals and software already available fit into the system. That way, the system will grow with you. And you'll never get left behind.

Based around the Z80A microprocessor, and utilising Microsoft™ BASIC, Aquarius™ has 8K ROM and 4K RAM resident within its console. It is able to provide up to 16 colours and resolution of 320 x 192, and generates its sound directly through the television's speakers.

With twin cartridge ports, the mini-expander allows simultaneous use of additional RAM and software cartridges. Twin disc game hand controls are included and the unit provides two additional sound channels. The 16K RAM cartridge plugs into either the console or the mini-expander, increasing Aquarius™'s RAM capacity to 20K.

With the ability to reproduce the entire graphic and character set of Aquarius™ at 80 characters a second, the printer's 40 column output allows transcription of the complete monitor image.

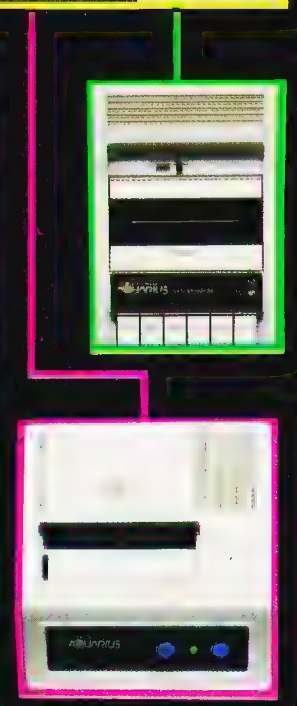




Using standard audio cassettes, the data recorder provides storage for programs and information, and allows the use of cassette based software. Incorporating a digital tape counter and transmission indicator, it operates sequential searching.

A large number of games, designed to take advantage of Aquarius™'s sophisticated colour and sound capabilities, are available on cartridges that plug into the console either direct, or through the mini-expander. Cassette based games can be used via the data-recorder.

A wide range of preprogrammed cartridges is available, including the LOGO teaching program and practical home data systems like FILEFORM™ and the spreadsheet calculator package, FINFORM™.



AQUARIUS™
HOME COMPUTER SYSTEM

WITH AQUARIUS, YOU WON'T GET LEFT BEHIND.

MATTEL ELECTRONICS®

WRITING ADVENTURES Pt. 2

For the purposes of this article we shall be working in BASIC. This will allow us to discuss the principles involved as generally as possible but it must be pointed out that BASIC is not an ideal language for this purpose because of its speed. Or rather its lack of speed. To program a BASIC Adventure and not bore your intended player, all of the tricks you can use to speed up program execution must be pulled out of the hat. Having an implementation of BASIC which is inherently fast, such as BBC BASIC, is a decided advantage too. However, for a truly fast response there is really little alternative but to resort to machine code. Although a poorly-written machine code program can still have a player drumming his fingers on the desk top.

INITIALISATION

Obviously the first job is to set up any arrays you may be using during the program. Adventures have the basic ingredients of places, directions to travel in, and objects to be carried by the player. However you decide to structure your program, you'll need at least one array. For example, the easiest way to keep track of the objects in a game is to store their locations in an array. A second (string) array could store their names. Another possibility is to use a string array to store the descriptions of the locations.

To speed things up later on, it might be an idea to assign values to any of the frequently-used variables at this point, in the order of their frequency of occurence. This places them at the beginning of the variable storage space in the computer's memory and so it takes less time to find them each time they are used.

Even more important is to use integer variables as far as possible. This allows the computer to store the variables in fewer bytes and also speeds up execution speed dramatically. Owners of micros such as the TRS-80 can do this globally at the start of the program with the DEFINT statement. Otherwise I'm afraid it's necessary to put the integer declaration on each variable, generally a % sign as in I%. This is tedious but helps to make a program playable.

On some computers you are required to set up a sufficient area of memory for string variables using the CLEAR statement. This should be followed by a number (the amount of string space you need) and for an Adventure game this could become quite large. Text is the main memory gobbler in this sort of program.

Once the housekeeping has been done, you can print the initial instructions on the screen, setting out the nature of the game, the objective and the rules — for example, what type of player response is required. Try to be brief: memory is at a premium! A 'press any key to continue' routine can be used between screenfuls if you can't get it all on at once.

Spectrum owners can jazz things up during the program load, of course, by including a 'title frame' to be displayed during loading using SCREEN$. Most commercial games for the Spectrum now do this.

LOCATION DISPLAY

Now that the system is set up, and the instructions digested, you now go to the initial location and tell the player where he is. The player's current location number is given by the variable LN, to which you should already have assigned the starting value for the reason described above. LN is used to select the correct text description to put on

There are several ways of storing text in a program. One way is to use a string array, L$( ), using one array element per location to store the description. Strings are limited on most machines to 256 characters, so the complete description of a location cannot be greater than this. This is no bad thing: if you had an Adventure with 80 locations and used the full 256 characters for each description, that would be 20K of memory gone in text alone. Using this method, the simple subroutine:

```
200 CLS
210 PRINT L$(LN)
220 RETURN
```

will put the correct description on screen. It will be necessary to insert padding spaces between words as needed, of course, to prevent words breaking at the end of a row. Each padding space is one less text character, too.

The disadvantage of this method is, how do you get the text into the string array? If you do it from within the BASIC program with assignment statements, for example:

```
1000 LET L$(1)="XXXXXXX"
1010 LET L$(2)="YYYYYYY"
```

and so on, the text is being stored twice in memory: once in the actual program, and once in the array area. So you lose twice as much memory as you need to. One way out of this problem is to dimension the array as before, then include a small routine in the initialisation section to read the text off a tape as a data file and load it into the array. Listing 2 gives an example for the MZ-80K. You will need to execute whatever file opening and closing statements are required by your version of BASIC. You can extend this to loading in the object descriptions into the array OB$ from tape.

Getting the data onto the tape is easy: you just write a dummy program that sets up the text arrays and saves them to tape as a data file. (Do remember that the save and load routines must have identical loop structures or you'll load the descriptions into the wrong array elements!) This system is best with a computer that has both motor control of the cassette drive and an auto-run feature. The main program loads, switches off the cassette, runs itself, dimensions the arrays and then turns on the cassette to load the data, which you have stored on tape immediately after the BASIC program.

A second method of getting text on to the screen is not to use the array L$()

```
10 CLEAR
19 REM ** CLEAR SHOULD BE FOLLOWED BY A NUMBER
20 DEFINT A-Z
30 DIM OB(100), OB$(100), L$(80), WD$(1)
40 LN=1: WC=0: TN=0
49 REM ** THESE VARIABLES EXPLAINED LATER
50 CLS: PRINT "WELCOME TO DEATHWORLD"
59 REM ** FOLLOW TITLE WITH ANY INSTRUCTIONS YOU FEEL NECESSARY.
```

Listing 1

```
60 ROPEN
70 FOR I=1 TO 80
79 REM ** ASSUMING 80 LOCATIONS
80 INPUT/T L$(I)
90 NEXT I
99 REM ** YOU COULD ALSO LOAD THE OBJECT DESCRIPTIONS LIKE THIS
100 FOR I=1 TO 40
109 REM ** ASSUMING 40 OBJECTS
110 INPUT/T OB$(I)
120 NEXT I
130 CLOSE
```

Listing 2

at all, but to have a series of PRINT statements in the program. This method uses the location number LN to choose which PRINT statements are executed. The ON-GOTO or better still, ON-GOSUB statements are obvious control structures in this case — although with a slight complication.

Most BASICs have restricted line lengths, and you won't be able to get all the required line numbers after the ON-GOSUB statement. Hence a bit of 'pre-sorting' is required. Suppose you have 80 locations and you can only get 27 line numbers on each ON-GOSUB program line. Listing 3 shows a suitable selection procedure.

The advantage here is that descriptions longer than 256 characters may be used simply by putting extra PRINT statements between the first one and the final RETURN. In the example shown, there is space for an extra eight lines of PRINT statements for each location, and by changing the line numbering this could be increased as necessary. Remember your memory, though — don't get carried away!

The CLS instruction is included for tidyness, since I'm not that keen on scrolling displays (although they do have the advantage that you can review previous moves as long as they remain on the screen). If you prefer to scroll, just leave out the CLS.

If your BASIC allows expressions to be used for line references, then life is much easier. Listing 4 shows how to program the screen display in this case. A further refinement used in some Adventure games is to give the full description of a location the first time a player enters it, and then abbreviated descriptions on the subsequent visits unless the player asks to look around. Listings 5, 6 and 7 show how this might be achieved using an array AB(80) to flag abbreviated locations for each of our methods of display. At the start of the game all the elements of AB( ) are zero. The PRINT system is starting to look a bit unwieldy now, unless you have GOSUB expression in your BASIC, but it is workable.

Now you will need to display any objects which have been left at that location. The descriptions of the objects are stored in the string array OB$( ), and their locations are in array OB( ). So if the first object is a knife, and it's in location 21, the OB$(1) is "a knife" and OB(1) = 21. Hence it is simple to print out all of the required objects, using a routine such as Listing 8. This simply checks through the array OB( ) and prints out the corresponding OB( ) element if OB( )=LN, the current location number.

This method will print each object on a new line, as in most commercial Adventures, which could lead to scrolling if you decided to leave everything in one plce. So word-processor-style wordwrapping might be a better idea, as shown in Listing 9. Here we count how far along the line we are before printing (using LL), and go to the nect line if a text string would be broken in the middle. Otherwise we can print on the same line. TV, a temporary variable, is used in either case to prevent anything from being printed if no objects are present.

Incidentally, there are two special values of LN used to register the position of an object. If the player is carrying an object, it is given the location number 0 (that's why we didn't use it to label a location last month), while 'hidden' objects which have not yet been brought into play are given an 'impossible' number, that is LN is larger than the largest location number. In our 80-room example, LN could be 81 for hidden objects.

PARSING THE INPUT

Now the player knows where he is and has to do something about it. To keep things simple we will limit him to a one- or two-word response, though there is no reason why a competent programmer shouldn't allow more complex constructions (remember the Hobbit!).

Let IN$ be the input string from the player. A suitable parsing routine has to decide how many words it contains, what they are, and whether it makes sense. Listing 10 shows how things might work. The input string is scanned through letter by letter (using LE$) and the first word is built up in WD$(0). When a space is detected, the first word is finished and the procedure continues to build up the second word in WD$(1). If another space is encountered, the procedure terminates and prints an error message.

The program makes a jump in the special case of one- or two-letter words (ie abbreviated commands such as I for Inventory or N for Go North), otherwise it tries to match the input words with verbs and nouns it

```
200 CLS
210 ON 1+INT(LN/27) GOSUB 230,250,270
220 RETURN
230 ON LN GOSUB 300,310,320, . . . , 540,550
240 RETURN
250 ON LN GOSUB 560,570,580, . . . , 810,820
260 RETURN
270 ON LN GOSUB 830,840,850, . . . , 1080,1090
280 RETURN
300 PRINT "DESCRIPTION OF LOCATION 1"
309 RETURN
310 PRINT "DESCRIPTION OF LOCATION 2"
319 RETURN
1090 PRINT "DESCRIPTION OF LOCATION 80"
1099 RETURN
```

Listing 3

```
200 CLS
210 GOSUB 290+10*LN
220 RETURN
300 PRINT "DESCRIPTION OF LOCATION 1"
309 RETURN
310 PRINT "DESCRIPTION OF LOCATION 2"
319 RETURN
1090 PRINT "DESCRIPTION OF LOCATION 80"
1099 RETURN
```

Listing 4

```
200 CLS
210 IF AB(LN)=1 THEN PRINT "YOU ARE IN THE ";
220 IF AB(LN)=1 THEN PRINT L$(LN,1): RETURN
229 REM ** SECOND SET OF ARRAY ELEMENTS CONTAIN SHORT DESCRIPTIONS
230 PRINT L$(LN,0): AB(LN)=1: RETURN
239 REM ** FIRST SET OF ARRAY ELEMENTS CONTAIN LONG DESCRIPTIONS
```

Listing 5

```
200 CLS
210 ON 1+3*AB(LN)+INT(LN/27) GOSUB220,230,240,250,260,270
215 RETURN
220 ON LN GOTO 300,310,320, . . . ,550
230 ON LN GOTO 560,570,580, . . . ,820
240 ON LN GOTO 830,840,850, . . . ,1090
250 ON LN GOTO 2000,2010,2020, . . . ,2250
260 ON LN GOTO 2260,2270,2280, . . . ,2520
270 ON LN GOTO 2530,2540,2560, . . . ,2790
300 PRINT "LONG DESCRIPTION, LOCATION 1": RETURN
1090 PRINT "LONG DESCRIPTION, LOCATION 80": RETURN
2000 PRINT "SHORT DESCRIPTION, LOCATION 1": RETURN
2790 PRINT "SHORT DESCRIPTION, LOCATION 80": RETURN
```

Listing 6

```
200 CLS
210 GOSUB 290+10*LN+AB(LN)*1700
220 RETURN
300 PRINT "LONG DESCRIPTION, LOCATION 1": RETURN
1090 PRINT "LONG DESCRIPTION, LOCATION 80": RETURN
2000 PRINT "SHORT DESCRIPTION, LOCATION 1": RETURN
2790 PRINT "SHORT DESCRIPTION, LOCATION 80": RETURN
```

Listing 7

recognises. These can be stored in DATA statements as the first three (usually) letters only. This saves space while providing enough scope for individual words. If you wish you can make the letter groups four or more letters long, but this is rarely necessary. If you have two words like BOAT and BOARD (both of which would be stored as BOA and hence be indistinguishable), you could simply change BOARD to PLANK. Of course that rules out the use of the words PLANE or PLANS or PLANET or PLANT in the same Adventure, but you get the idea!

```
2800 TV=0
2810 FOR I=1 TO 100
2819 REM ** ASSUMING 100 OBJECTS
2820 IF OB(I)=LN THEN TV=1: I=101
2830 NEXT I
2840 IF TV=0 THEN RETURN
2850 PRINT "THERE IS "
2860 FOR I=1 TO 100
2870 IF OB(I)=LN THEN PRINT OB$(I)
2880 NEXT I
2890 PRINT "LYING HERE"
2900 RETURN
```

Listing 8

```
2800 TV=0: LL=9
2809 REM ** LL = LINE LENGTH COUNTER, SET TO LENGTH OF "THERE IS"
2810 FOR I=1 TO 100
2820 IF OB(I)=LN THEN TV=1: I=101
2830 NEXT I
2840 IF TV=0 THEN RETURN
2850 PRINT "THERE IS ";
2860 FOR I=1 TO 100
2870 IF OB(I)<>LN THEN 2900
2880 IF LEN(OB$(I))+LL>40 THEN PRINT: LL=0
2889 REM ** ASSUMES LINES ARE 40 CHARACTERS LONG - CHANGE TO SUIT
2890 LL=LL+LEN(OB$(I)): PRINT OB$(I);" ,";
2900 NEXT I
2910 IF LL+10>40 THEN PRINT
2920 PRINT "LYING HERE"
2930 RETURN
```

Listing 9

```
3000 INPUT "WHAT DO YOU WANT TO DO "; IN$
3010 WC=0: WD$(0)="": WD$(1)=""
3020 FOR I=1 TO LEN (IN$)
3030 LE$=MID$(IN$,I,1)
3040 IF LE$="[SPC]" THEN WC=WC+1: GOTO 3060
3050 WD$(WC)=WD$(WC)+LE$
3060 IF WC=2 THEN I=LEN(IN$)+1
3070 NEXT I
3080 IF WC=2 THEN PRINT "ANSWER IN ONE OR TWO WORDS PLEASE": GOTO 3000
3090 IF WC=0 AND LEN(WD$(0))<3 THEN 3280
3100 RESTORE: AN=0: VE$=LEFT$(WD$(0),3)
3110 READ TS$
3120 FOR I=1 TO LEN(TS$) STEP 3
3130 TW$=MID$(TS$,I,3)
3140 IF TW$=VE$ THEN AN=101*I: I=LEN(TS$)+1
3150 NEXT I
3160 IF TW$<>"XXX" AND AN=0 THEN 3110
3170 IF TW$="XXX" THEN PRINT "I DON'T KNOW HOW TO ";WD$(0): RETURN
3180 IF WC=0 THEN RETURN
3190 TV=0: NO$=LEFT$(WD$(1),3)
3200 READ TS$
3210 FOR I=1 TO LEN(TS$) STEP 3
3220 TW$=MID$(TS$,I,3)
3230 IF TW$=NO$ THEN AN=AN+I: I=LEN(TS$)+1
3240 NEXT I
3250 IF TW$<>"ZZZ" AND TV=0 THEN 3200
3260 IF TW$="ZZZ" THEN PRINT "I DON'T KNOW WHAT A ";WD$(1);" IS": RETURN
3270 RETURN
3280 REM ** SERIES OF 'IF VE$="XX"' STATEMENTS TO JUMP TO INVENTORY (I),
3290 REM ** HELP (H), MOVEMENT (N, NE, W, UP, D ETC) ROUTINES.
50000 DATA GETDROTAKRUNKILJUMEATSWICLISHOUSEFIRXXX
50009 REM **GET,DROP,TAKE,RUN,KILL,JUMP,EAT,SWIM,CLIMB,SHOOT,USE,FIRE
50010 DATA GUNKNIWINBREBOASWOPLAROPCLOBOTOARLAD
50019 REM ** GUN,KNIFE,WINE,BREAD,BOAT,SWORD,PLANK,ROPE,CLOAK,BOTTLE,OAR,LADDER
50020 DATA BOXJEWCLITREBIRNESDOOROOCORROCCAVZZZ
50029 REM ** BOX,JEWEL,CLIFF,TREE,BIRD,NEST,DOOR,ROOM,CORRIDOR,ROCKS,CAVE
```

Listing 10

For testing purposes VE$ is assigned the first three letters of WD$(0), so that we don't have to waste time using LEFT$ on each test. This also leaves WD$(0) intact should we need to printbit out in an error message later. Line 3110 to 3150 check the three-letter groups from the DATA statements against VE$ and if a match is found, the 'action number' is set to 101 times the loop counter. (The significance of 101 is explained later). By making the last triplet in a group of DATA statements XXX, we can have as many stored words as we want and line 3160 ensures we loop back until they're all tested.

The subroutine terminates if no match is found, or if the input consisted of a single word. Otherwise a similar loop matches the second word, WD$(1), against a set of noun triplets. The index of the matching triplet is added to the action number.

This routine is not exhaustive in its error-checking (what happens if a valid word has only two letters in it? — left as an exercise for the reader), but it provides a good starting point for the more sophisticated systems.

One final point about word recognition — even given the constraints of memory, it's a good idea to design your program so that the parsing routine will recognise any of the nouns used by the program itself in its location descriptions, even if they play no part in the Adventure. I'm fed up with poorly-written commercial games which tell you "You're in a room . . ." and when you ask EXAMINE ROOM, reply "I don't know what a room is". This just looks inept: and it's an example taken from the supposed King of Adventures, Scott Adams. This is where I commit he ultimate heresy and go on record as stating that I don't think much of Adams' programs. The plots can get very silly in places (as opposed to humorous), and when a friend made a complete disassembly of the Z80 version of "The Count", he found several errors and sections of inefficient and pointless programming. I'm not sure why other Adventure writers rave about him so much — being first doesn't necessarily mean being best. Oh, well . . .

THE ACTION NUMBER

The variable AN returned by the parsing routine is a simple way of coding any verb-noun pair recognised by the program into a unique number. Suppose you have V verbs and N nouns. Then $(V^*(N+1))+N$ is a unique number for any combination of numerical index of verb and noun. Take our word for it if you're not mathematically-minded: in a way you're counting in base N. This system gives a fair amount of scope regarding the numbers of verbs and nouns in your program. For example, if you're working in integer arithmetic for speed, the average eight-bit micro has a maximum positive integer of 32767. Now (181 x 180)+180 is 32760, so you could have a game with 180 different verbs and 180 different nouns and still not generate an action number that exceeds the integer limit. We used 101 in our routine because our example program has 100 nouns.

Given the action number, and a series of flags to signal any special conditions (for example, SD = 0 if a secret door is shut, SD = 1 if it's open), then the player's inventory, location, action number and flags can be combined in tests to determine what action, if any, results from the user input. The simplest method is just a long list of IF-(AND-OR)-THEN tests, but things can be speeded up a bit by using ON-GOTO or ON-GOSUB in a similar fashion to the location display routine. You could sort by action number range, or by location.

MOVEMENT

There are several ways of coding movement routines in Adventure games. One of the simplest is to have a set of direction variables — for example, N, S, E, W, U and D — and assign them new values depending on which location you're currently standing in. The values assigned, of course, are the location numbers of the places you arrive at if you travel in the corresponding direction. A value of zero in any of the variables means that there is a dead end in that direction: you cannot go that way.

Movement itself now consists of checking the direction variable which corresponds to the route the player wishes to take, altering LN to the value of that direction variable if it's non-zero, and printing "You can't go that way" if it is. If control is passed back to the location display subroutine, the new value of LN results in a description of the new location to be displayed on the screen.

Another way of storing the prmitted directions in which to move would be to have a string array DI$( ) dimensioned to the number of locations. The entry for each array element would be a string consisting of the destination given by each direction, which could be extracted to obtain the new location number by using MID$ and VAL. A typical entry might be "002300000024", meaning that, for example, going south leads to location 23, going down to 24, and all other exits are barred. The right number is given by VAL(MID$(DI$(LN),I,2)), where I is the index for the direction chosen by the player (you could get this from a look-up table). The extra string operations would make this a bit slow in BASIC but the idae could be useful if adapted for a machine code program.

GET, DROP AND INVENTORY

These are quite simple to implement. GET OBJECT simply checks that the object is in the current location and that any special conditions are fulfilled (the classic example being that you cannot get the bird unless you have the cage and drop the rod!), then sets the location number to zero. DROP checks that the player is actually holding the object and sets its location number to that of the current location, LN. The command INVENTORY (I), just checks through the array OB( ) and prints out "You are carrying " followed by any string in OB$( ) corresponding to a location number of zero in OB( ). In fact the routines are similar to those used by the location display subroutine.

DO-IT-YOURSELF

Given these ideas, any enterprising programmer can get a reasonable Adventure running — it should be obvious now how routines such as LOOK, QUIT, SCORE or the game ending can be programmed. Don't forget a game save feature to store the current values of all the variables: it's essential in a long game. And as we pointed out at the start, these are only general principles, and there's no reason why you can't find more sophisticated, more efficient and more rapid methods of doing anything we've covered. To give you a little more incentive, we've come up with the following competition . . .

COMPETITION

We want to make one of our readers rich. And famous. All you have to do is write an Adventure game for us to play. (Who says the life a journalist is one of mindless tedium).

Apart from that there are very few limitations. Any subject matter is fair game, but bear in mind we've played an awful lot of Adventures between us here at Computing Today, and plagiarists should be warned that we'll almost certainly spot their efforts — and disqualify them.

The prize for the author of the game considered to be the best by our panel of judges is to have his game marketed by our sister software company, Argus Press Software Ltd, and receive full royalties on all sales. That could amount to a fair income for a popular game, particularly as we will have the game converted to run on a range of popular machines.

To enter this competition, you should submit your game on tape, stating clearly what machine it runs on, together with your name and address and a sealed envelope containing a map and solution to the Adventure (in case we get stuck!). Make sure your name, address and the name of the game is marked clearly in block letters on the cassette case and envelope.

We can accept programs for the following machines on cassette:
Commodore 64, VIC 20, both PETs;
ZX Spectrum;
Dragon 32;
Any Atari;
Oric-1;
BBC Micro;
Texas TI99/4A (with or without Extended BASIC);
Sharp MZ-80K and MZ-80A (with standard BASIC only);
TRS-80,
Apple II or IIe;
Microtan 65;
Nascom 3;
Cortex.
In addition we will accept programs on disc for the Commodore computers, the Ataris, both Sharps and the Apples.

The majority of the program must be written in BASIC. Specific routines such as screen-handling can be in machine code but BASIC equivalents should be provided where possible. Machine code routines must be relocatable and fully documented. (This is in your interest: it makes it easier to convert programs onto other machines and the more we can adapt it to, the more tapes we sell and the more royalties the winner earns.)

The closing date for the competition is January 31st 1984. The authors of programs which do not win will retain full rights in their work.

RULES OF ENTRY

This competition is open to all UK and Northern Ireland readers of Computing Today except for employees of Argus Specialist Publications Ltd, their printers, distributors or anyone associated with the competition.

All entries must be postmarked before the closing date of January 31st 1984.

No correspondence will be entered into regarding the result of this competition and it is a condition of entry that the judges' decision is final.

The winner will be notified by post and the result of the competition will be published in a future issue of Computing Today.

Entries should be marked clearly on the outside of the envelope "ADVENTURE COMPETITION" and addressed to Computing Today at the address on page 3.


WHATEVER HAPPENS
IN THE FUTURE, IT'LL
FIT INTO THIS SPACE.
COPY
FOR
NEXT
POINT
POKE
PEEK
VAL
RTN
MATTEL ELECTRONICS®
AQUARIUS™

WHEN IT COMES TO PERSONAL COMPUTER SPECTRAVIDEO IS BEYOND ANY COMPARE!

To the delight of our customers, and the dismay of our competitors, we now proudly present the most advanced, most capable personal computer system available today: Spectravideo™'s SV-318 and SV-328. While the SV-318 is "everything home computer users were waiting for," "the SV-328 is specially designed "for the small business that doesn't plan to stay that way." These 2 computers, too, are beyond any comparison in their price range.

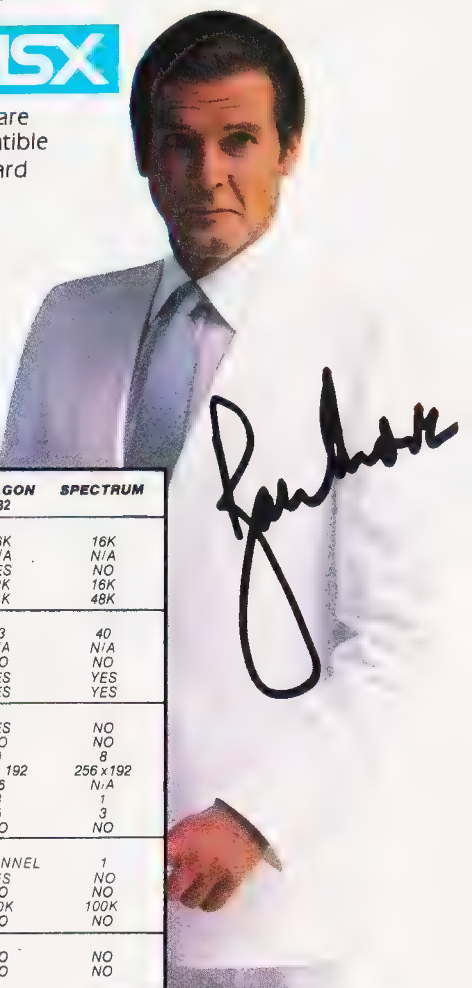
And they do not stand alone. Seldom, if every, has a new computer been supported by so many peripherals. Of exceptional quality, all this hardware rides into you on the crest of the very last technology.

Additionally — and importantly — Spectravideo™ is very software oriented. With built-in CP/M compatibility, the SV system allows you to take advantage of thousands of pre-existing programs. And with the MSX software compatible standard, jointly launched by Spectravideo™ and most of Japan's largest electronics firms, Spectravideo™ can take advantage of all software developed by other MSX participants. Plus, Spectravideo™ is now producing its own line of top-quality software. Finally, the SV system includes several innovative and interesting accessories that you will want to use.

Because of product depth and ease-of-operation, the SV system will give you full usage from the moment you unpack it. Yet it is both capable and expandable enough to give you long-term usage, too. That's why Spectravideo™ is truly, "The computer system you'll grow into, not out of."

MSX

software compatible standard



	SPECTRAVIDEO SV 328	SPECTRAVIDEO SV 318	APPLE II E	ATARI 800	COMMODORE 64	BBC MODEL B	DRAGON 32	SPECTRUM
COMPUTING POWER FEATURES								
BUILT-IN ROM	**48K**	**32K**	16K	10K	20K	16K	16K	16K
EXPANDABLE TO	**96K**	**96K**	N/A	42K	N/A	64K	N/A	N/A
BUILT-IN EXTENDED MICROSOFT® BASIC	**YES**	**YES**	YES	ADDITIONAL COST	NO	NO	YES	NO
BUILT-IN RAM	**80K***	**32K*****	64K	48K	64K	32K	32K	16K
EXPANDABLE TO	**256K****	**256K****	64K	NO	N/A	32K	64K	48K
KEYBOARD FEATURES								
NUMBER OF KEYS	**87**	**71**	63	61	66	73	53	40
USER DEFINE FUNCTIONS	**10**	**10**	N/A	4	8	10	N/A	N/A
SPECIAL WORD PROCESSING	**YES**	**YES**	NO	NO	NO	NO	NO	NO
GENERATED GRAPHICS (FROM KEYBOARD)	**YES**	**YES**	NO	YES	YES	YES	YES	YES
UPPER/LOWER CASE	**YES**	**YES**	YES	YES	YES	YES	YES	YES
GAME/AUDIO FEATURES								
SEPARATE CARTRIDGE SLOTS	**YES**	**YES**	NO	YES	NO	NO	YES	NO
BUILT-IN JOYSTICK	**NO**	**YES**	NO	NO	NO	NO	NO	NO
COLORS	**16**	**16**	15	128	16	16	9	8
RESOLUTION (PIXELS)	**256x192**	**256x192**	280 x 160	320 x 192	320 x 200	256 x 640	256 x 192	256 x 192
SPRITES	**32**	**32**	N/A	4	8	N/A	16	N/A
SOUND CHANNELS	**3**	**3**	1	4	3	1	3	1
OCTAVES PER CHANNEL	**8**	**8**	4	4	9	3	5	3
A.D.S.R. ENVELOPE	**YES**	**YES**	NO	NO	YES	YES	NO	NO
PERIPHERAL SPECIFICATIONS								
CASSETTE	**2 CHANNEL**	**2 CHANNEL**	1 CHANNEL	2 CHANNEL	1 CHANNEL	2 CHANNEL	2 CHANNEL	1
AUDIO 1/0	**YES**	**YES**	NO	YES	NO	NO	YES	NO
BUILT-IN MIC	**YES**	**YES**	NO	NO	NO	NO	NO	NO
DISK DRIVE CAPACITY	**256K**	**256K**	143K	92K	170K	100K	100K	100K
(LOW PROFILE)	**YES**	**YES**	NO	NO	NO	NO	NO	NO
CP/M® COMPATIBILITY (Standard 80 column programs)								
CP/M® 2.2	**YES**	**YES**	NO****	NO	NO*****	YES	NO	NO
CP/M 3.0	**YES**	**YES**	NO	NO	NO	NO	NO	NO

Specifications are subject to change without prior notice

* 64K user addressable plus 16K graphic support

** 240K user addressable plus 16K graphic support

*** 16K user addressable plus 16K graphic support

**** Apple II can accept modified 40 or 80 Column CP/M

***** Commodore 64 accepts 40 column CP/M

For further information contact:
C K SUPPLIES
UNIT 5, NORSIDE, OLDMIXON CRESCENT,
WESTON-SUPER-MARE, AVON.
TEL: (0934) 418838/516246

SPECTRAVIDEO™

Paul Gardner

LEARNING FORTH PART 2

Having introduced the stack and the definition of FORTH words, we now extend our ideas a little further. In case we need to change our minds, variables make an appearance too.

Last month I explained briefly how to define a new FORTH word whichcould be added to the dictionary. To recap and to clarify a few basic points, each new word can be defined in the form

```
: name  -body of definition- ;
```

where that 'name' is the name of the word.

Note that new definitions can be edited into a disc block and subsequently loaded or can be compiled from the keyboard. Some versions of FORTH will execute the input buffer immediately if you type more than 80 characters before pressing Enter. This can cause problems when entering long definitions at the keyboard so it is safer to edit them onto a disc block first.

The body of the definition consists of the instructions to be performed whenever the new word is executed. A definition can contain different types of instructions:

- In its simplest form a new word contains a list of previously defined words, for example

```
: CLEANSCREEN CLS CR CR CR ;
```

When executed CLEANSCREEN will clear the screen and perform three carriage returns so that the next item to be printed will appear on the fourth line.

- A definition can contain an item to be printed using the FORTH word ." (pronounced dot-quote), thus

```
: TITLE CLEANSCREEN ." this is an example" CR ;
```

You should ensure that the ." is separated from the string to be printed by at least one space. (It doesn't matter if the final " is next to another character because of the way ." works.)

- Comments can be included in a definition, though this is only necessary when you are editing a definition onto a disc block to be compiled later. This example should be edited onto a disc block and then compiled.

```
: TITLEPAGE ( clears the screen and
  prints a heading for the page on
  the fourth line)
  CLEANSCREEN
  ." Official document heading 1A " CR ;
```

(is a FORTH word just like ." and must be separated from the comment by at least one space. (Have you noticed yet the problem of distinguishing FORTH words from punctuation in text? We're doing our best to structure sentences so that it's obvious what we mean.)

It is useful in the development of large FORTH programs to add numerous comments: they are ignored by the compiler during compilation and so unlike BASIC REM statements they do not take up any precious memory space.

- Numbers can be included within a definition. When the new word is executed any number contained in the definition will be put on the stack before it is used, for example

```
: PRICE ( puts a number on the stack) 130 ;
```

If you execute PRICE the number 130 will be put on the top of the stack. (Remember you can print the top item on the stack using . (pronounced dot)). For the moment we will only consider numbers which are signed integers in the range +32767 to −32768.

Any new definition can contain any or all of the different types of instruction described. Here is a final example:

```
: DISPLAYPRICE
  TITLEPAGE CR
  ." According to our records your " CR
  ." computer costs £ " PRICE . CR ;
```

Remember your syntax:

- every : needs a ; to complete a definition
- every ." needs a " to end a string of text
- every (needs a) to contain a comment.

The type of definition described above is called a colon definition because it uses : but there are other sorts of definition, as we will see.

BACK TO THE STACK

As we have seen, the computer can be made to store numbers temporarily using the stack. This is obviously done using electronics but it can be imagined as described before, as a tube into which coins of different values can be placed. You can only readily get at the one on the top and the first coin in will be the last out.

Figure 1 shows this imaginary stack (which is spring-loaded to keep everything pushed to the top). The bottom line of the diagram gives the actions performed which

Fig. 1 The stack. The imaginary spring forces items to the top.

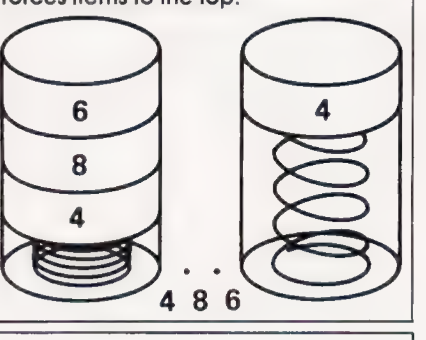
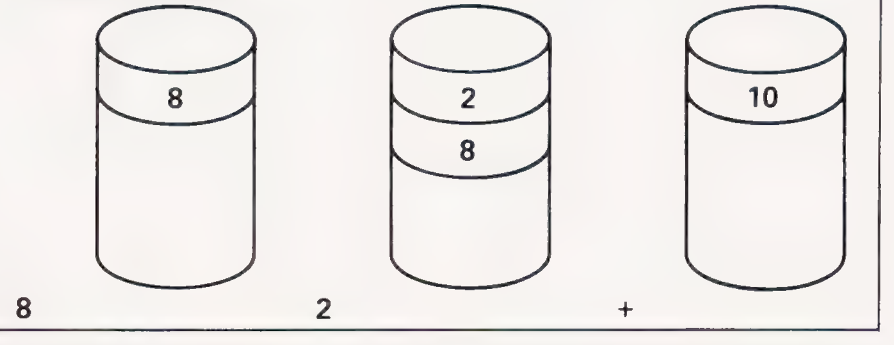


Fig. 2 The stack action when two numbers are added.

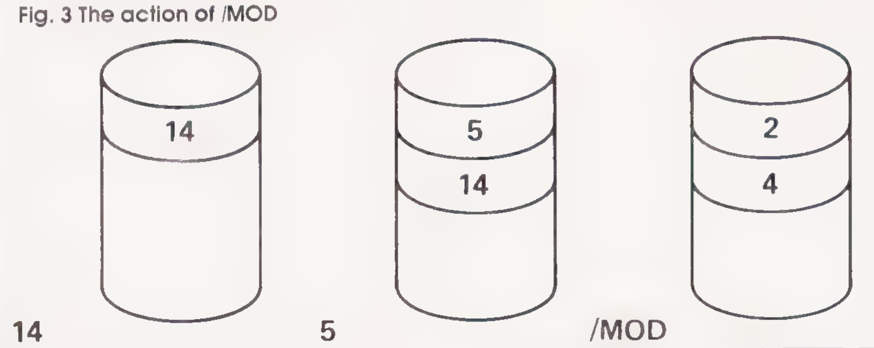


produce the stack contents shown, ie for Fig. 1 if you type in three numbers, 4 8 6, each number will be pushed onto the stack so that the stack contents will be as shown with the number 4 at the bottom and 6 on top. If you now type .(dot) twice, this will print the top two numbers and leave the number 4 at the top of the stack.

Figure 2 shows the action of the stack when two numbers are added together. When the word + (plus) is executed, at least two numbers are expected on the stack. The top two are added together by the computer and the result is put back on top of the stack.

Figure 3 shows that particular operators may leave more than one result. /MOD (slash-mod) takes two numbers off the stack, divides the bottom one by the top and leaves both the quotient and the remainder on the stack as results.

Fig. 3 The action of /MOD



Rather than using pretty diagrams, FORTH programmers usually use a common notation to describe the stack effects of a FORTH word. The basic form is:

```
( before -- after)
```

The dash separates the things that should be on the stack before you execute the word from the things that will be left there afterwards. For example, the stack notation for the word . is shown below:

```
( n --)
```

(The letter 'n' stands for 'number'). This shows that . expects one number on the stack (before) and leaves no number on the stack (after). Here's the stack notation for the word /MOD we introduced earlier:

```
/MOD      ( n1,n2 -- remainder,quotient)
```

When there is more than one n, we number them n1, n2, n3 etc. The numbers 1 and 2 do not refer to the position on the stack. Stack position is indicated by the order in which the items are written; the ***rightmost*** item on either side of the dash is the topmost item on the stack. For example in the stack notation of /MOD above, the quotient is on top.

If you are writing the definition of a new word onto disc it is a good idea to include a description of the stack effects as a comment at the beginning of a new word. So for a word to cube the top number on the stack

```
: CUBE    ( n -- n*n*n)
  DUP DUP * * ;
```

There are numerous arithmetic operators available in FORTH. These should have their stack effects described in the manual accompanying your version. Most of the operators behave as expected (+, −, /, * and so on). However, as FORTH usually expects numbers to be integers in the range +32767 to −32768 there are a few words which behave a little oddly. Here are some of them:

MOD will produce the integral remainder after division and /MOD has already been described.

*/ (star-slash) takes three numbers off the stack and returns one. Its stack effect is:

```
*/        ( n1,n2,n3 -- n1*n2/n3)
```

ie it multiples n1 by n2 then divides this result by n3. The usefulness of this word is that the intermediate result can exceed the usual limit for numbers as it is stored as a double length number (more of this at later date). As long as the final result is in the usual range then */ produces the expected result.

*/MOD is an extension to this and leaves the remainder of the division as well as the quotient.

```
( LISTING 1 )
( STACK PRINT ROUTINE FOR )
( ABERSOFT fig-FORTH 1.1  )
( SPECTRUM MICRO          )

: DEPTH
  S0 @ SP@ - 2 - 2 /  ;

: .S
  CR DEPTH
  IF
    SP@ 2 - S0 @ 2 -
    DO
      I @ . -2
    +LOOP
  ELSE ." Empty " THEN ;
```

```
*/MOD     ( n1,n2,n3 -- remainder,n1*n2/n3)
```

Beginners who are just learning to manipulate numbers on the stack may often find themselves typing a series of dots to see what is on the stack after their manipulations. The problem with dots, though, is that they don't leave the numbers on the stack for future use. Listing 1 is the definition of a very useful word for beginners and experienced users alike. .S prints out all the values that happen to be on the stack 'non-destructively'; that is, without removing them. Type in the definitions of DEPTH and .S shown, and don't worry for the moment how it works. Test it first with nothing on the stack:

```
.S
Empty ok
```

Now try it with numbers on the stack:

```
1 2 3 .S
```

produces `1 2 3 ok`

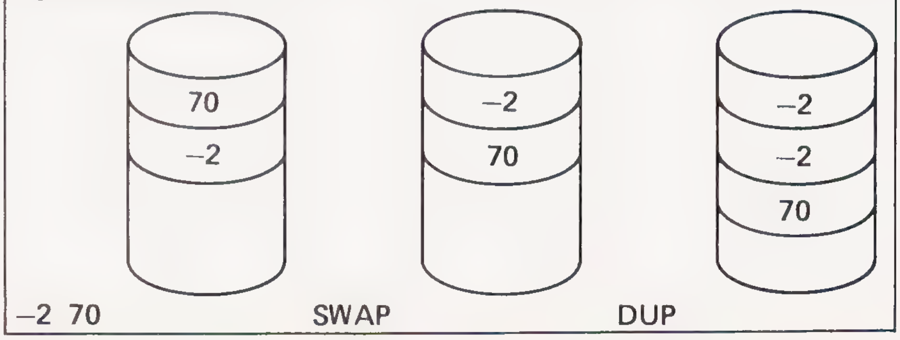
Now try `DUP .S`
It produces `1 2 3 3 ok`

This is a useful word for the next section.

STACK MANIPULATION

Some books use mathematical exercises to introduce the subject of stack manipulation, so I will too. For example, convert the following infix expression (ie not reverse Polish notation) into a FORTH definition which will yield a result, and show

Fig. 4 SWAP and DUP



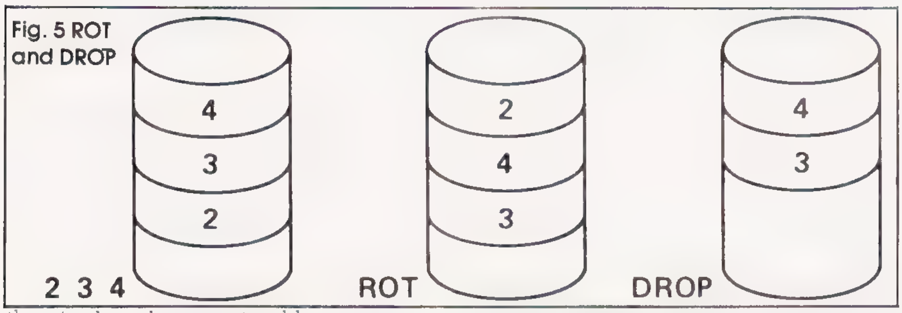


Fig. 5 ROT and DROP

the stack order required by your definition.

```
a - 4b + c
  6
```

A definition using three numbers a, b and c from the stack and leaving one result could be

```
: QUIZ ( c,a,b -- result)
  4 * - 6 / + ;
```

Easy isn't it? Another example might be:

```
a - b
  c
```

Well, you can't do this one unless you can rearrange the order of items on the stack in some way. So far we have met the word DUP which produces a copy of the top item on the stack. Next comes SWAP. So the answer to the above problem becomes:

```
: QUIZ1 ( c,a,b -- result)
  - SWAP / ;
```

As you've probably guessed, SWAP takes the top two items off the stack, swaps them around and replaces them.

SWAP (n1,n2 -- n2,n1)

Figure 4 shows the actions of both SWAP and DUP diagramatically.

There are several other stack manipulation commands:

OVER (n1,n2 -- n1,n2,n1)

Makes a copy of the second item and pushes it on top.

ROT (n1,n2,n3 -- n2,n3,n1)

This one (pronounced 'rote') rotates the top three items, bringing the third one to the top and pushing the other two downwards

DROP (n --)

Discards the top stack item (more useful than you might think). Figure 5 shows the diagramatic effects of ROT and DROP.

Two mind-bogglingly useful words that are missing from Abersoft's version of FORTH are PICK and ROLL.

PICK (n -- copy of nth item on stack)

Takes a number, n, off the top of the stack and makes a copy of the nth one down from the top of the stack in what remains, leaving this copy on top.

Fig. 6 The action of PICK

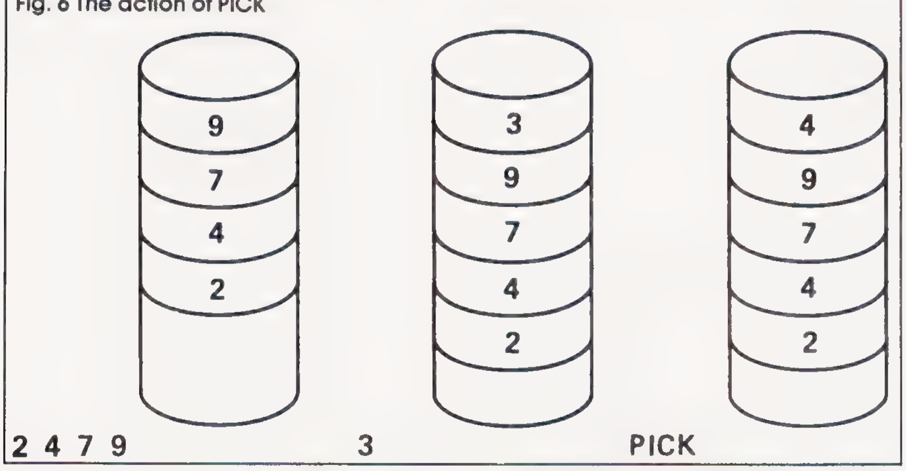


Figure 6 shows this better than an explanation. The left hand drawing shows a stack of four numbers. The expression 3 PICK will make a copy of the third item (in the initial stack) and push the copy onto the top.

ROLL (n --)

This takes a number, n, off the stack and then in what is left rotates the top n numbers, bringing the nth to the top.

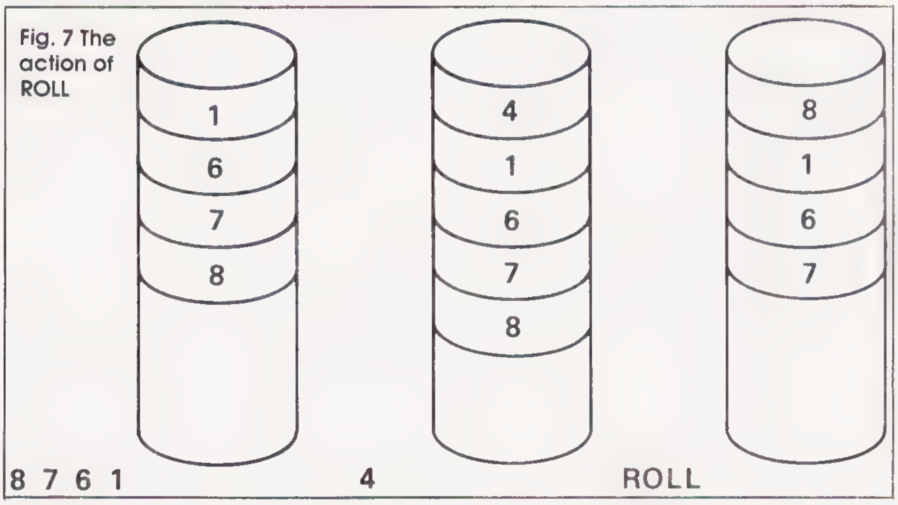


Fig. 7 The action of ROLL

Figure 7, again, shows this more clearly. The left hand drawing shows the initial stack. The expression 4 ROLL removes the fourth item (in the initial stack) and puts it on top. All the other elements are shuffled down one to fill the empty space. If you want a 'coin in the tube' analogy for ROLL, here it is: "Paul Daniels removes the top coin from the tube and reads the number, n, that you have scratched on it. He puts this coin in his pocket and then by some devilishly cunning sleight of hand manages to extract the nth coin from the tube and puts it back on the top."

```
( LISTING 2 )
( PICK ROUTINE FOR )
( ABERSOFT fig-FORTH 1.1 )
( SPECTRUM MICRO )

: PICK
 2 * SP@ SWAP + DUP S0 @ <
 IF
  @
 ELSE
  CR ." Below Stack "
  DROP QUIT
 THEN ;
```

```
( LISTING 3 )
( ROLL ROUTINE FOR )
( ABERSOFT fig-FORTH 1.1 )
( SPECTRUM MICRO )

: ROLL
 2 * SP@ + DUP S0 @ <
 IF
  DUP @ SWAP SP@ 2 + SWAP
  DO
   I 2 - @ I ! -2
  +LOOP
  DROP
 ELSE
  CR ." Below Stack "
  DROP QUIT THEN ;
```

Listings 2 and 3 give definitions for PICK and ROLL for use with the

Abersoft version of FORTH for the Spectrum. Note that PICK, ROLL and .S are such useful commands that it is well worth following the instructions in the manual to save these definitions as part of an extended dictionary (ie in their compiled form). Then they will always be present when you load FORTH initially.

In order to practice the use of FORTH's stack manipulation commands and the arithmetic operations, try making up a few complicated infix expressions and produce definitions to calculate the results.

VARIABLES, OR PLACES TO PUT NUMBERS

In an earlier example I used a word PRICE to leave a number on the stack whenever the word was executed. This works fine and is all very well, but what happens if the price changes? You cannot change the value that PRICE will leave on the stack once the word has been compiled, so FORTH provides a method of storing numbers in the dictionary as ***variables***. For example try typing in the following:

```
12 VARIABLE DOZEN
```

If you now try to VLIST you will see that DOZEN has been added to the dictionary, so what does it do? Try

```
DOZEN U.
```

You will probably see a fairly large number printed on the screen. U. prints the top number off the stack but assumes that it is a positive integer. Now type

```
DOZEN @ .
```

and 12 will be printed.

The number printed first is actually the memory address of the newly-formed variable DOZEN. The second number, 12, is the value stored in this variable. This is the number you initialised the variable with when it was defined. What happens whenever you type the name of a word defined using VARIABLE is that the memory address of the variable is left on the stack. There are two words in FORTH which commonly use this address:

```
@         ( addr -- n)
```

This word (pronounced 'fetch') takes a number off the stack and returns the number currently storedat that address.

```
!         ( n,addr --)
```

This word (pronounced 'store') takes a number and an address fom the stack and stores the number at the given memory address. As an example

```
13 DOZEN !
```

will change the value stores in DOZEN to 13. Now DOZEN @ . will produce a baker's dozen!

There are a few interesting points to note here. Unlike BASIC, where a new variable can be produced by a LET statement within a program, FORTH's variables have to be defined before they can be used. Also, if you take a look at the memory map of a BASIC system (page 121 of the Spectrum handbook) you see that variables are stored separately from the program. In FORTH, when a new variable is created it is added to the dictionary, and its value can only be changed explicity by using @ or ! since there is no FORTH equivalent to RUN or CLEAR to delete all the variables in one go.

For those familiar with PEEK and POKE in BASIC, then @ and ! are FORTH's equivalents but they fetch or store a 16 bit number using the address provided as the address of the first byte of the number. They are more closely equivalent to DEEK and DOKE (used in a few BASICs to access 16 bits of memory).

A variable such as DOZEN, once defined, can be used in another definition in the same way that it is used from keyboard. For example,

```
10 VARIABLE DATE
11 VARIABLE MONTH
1983 VARIABLE YEAR
: DISPLAYDATE CLS
  ." Today's date is "
  DATE @  .  ." :"
  MONTH @  .  ." :"
  YEAR @  .  CR ;
```

This should produce an output like:

```
Today's date is 10:11:1983
ok
```

Two useful words defined in many FORTH systems are, first:

```
?         ( addr --)
```

This takes an address off the stack and displays on screen the value stored there. So MONTH ? would print 11 on screen. ? is defined as

```
: ? @ . ;
```

Secondly, we have the word

```
+!        ( n,addr --)
```

This adds the number n to the value currently held at the address supplied.

```
1 MONTH +!
```

would add 1 to the value contained so that MONTH ? now produces 12. This allows us to use a variable as a counter.

Variables are useful for keeping a count of something that happens repeatedly. For example you could be running a bookshop in which you want to keep count of the number of different types of book you sell a day. First define

```
0 VARIABLE TBOOKS
0 VARIABLE FBOOKS
```

We'll imagine that you only sell two types, technical books (TBOOKS) and fiction books (FBOOKS). At the beginning of each day you want to reset the counters to zero. You could reset both variables in a definition like

```
: RESET 0 TBOOKS !
        0 FBOOKS ! ;
```

Then you put your keyboard next to the cardboard box where you keep the money (odd shop this!). You could define a couple of useful words to make life easier for you during the day.

```
: TBOOK 1 TBOOKS +! ;
: FBOOK 1 FBOOKS +! ;
```

Both these words add 1 to the variable associated with each type of book. Now each time you sell a book you key in either TBOOK or FBOOK to add one to the correct counter. At the end of the day,

```
TBOOKS ?
FBOOKS ?
```

will print the counts of the respective number of books sold.

Let's try it. (In this example the computer's responses are in bold).

```
RESET ok
TBOOK ok
FBOOK ok
TBOOK ok
TBOOK ok
FBOOK ok
TBOOKS ? 3 ok
FBOOKS ? 2 ok
```

For those without +! it is usually defined as:

```
: +! DUP @ ROT + SWAP ! ;
```

See if you can follow the stack effects during the execution of this word.

Important: words such as ? and +! are fundamental to the FORTH philosophy. By replacing commonly occuring sequences of instructions like @. with ? or actions such as

```
DATE @ 1 + DATE !
```

with

```
1 DATE +!
```

you effectively factorise your programs, which usually makes them shorter, faster and easier to follow.

Most of the examples we have met so far have been short and simple. Each definition, when executed, has been followed from beginning to end without any control of flow. Next month we will see how sections of a definition can be repeated or made optional by using structures such as DO -LOOP and IF - ELSE - ENDIF. We shall also see how you can create your own data types by examining how definitions are stored in the computer's memory.

sinclair special

5

Inside...
New Interface 2 and ROM cartridges!
New Software!

TAKING NEW SOFTWARE IN NEW DIRECTIONS

You'll see that this issue of Sinclair Special devotes considerable space to software. Why, when we've so much to say about hardware and peripherals? Simply because at Sinclair we believe in supporting first-class hardware with first-class software.

This month sees the start of a new commitment to education in our catalogue, both for adults and children.

In the field of micro theory, we've programs like Beyond BASIC and Make-a-Chip, which take you from the creation of simple ZX® assembler subsets to simulated circuit design projects.

There's Musicmaster, to teach you music terminology, note values and composition.

And if you're keen to beat your Spectrum at chess (which can be hard), you'll certainly want to try Chess Tutor 1, the first program in a complete chess masterclass.

Coming soon...

In the pipeline are many new releases, some of which break completely new ground. LOGO and micro-PROLOG for instance. They're fifth generation languages which will take you and your Spectrum closer than ever before to the creation and application of artificial intelligence.

A formal agreement between Sinclair and Macmillan Education has been announced, the first results of which will be published this autumn. These consist of five programs in a complete early reading course plus the first four of a series of programs based on Macmillan's top selling Science Horizons Scheme. All programs are designed for use in schools or the home.

And with Blackboard software, we're publishing six more home education programs for primary school children. Covering alphabet, spelling and punctuation, each of these programs is a true gem, unlike any other education software, and fascinating to run. Even for adults!

I believe that these new titles represent a major advance in educational software for the home.

New ROM software too!

You may well have heard news of ZX Interface 2® and ROM cartridge programs. You'll find full details of the Interface and its software on the facing page (and there's an order form on the back page too!). These offer an instant games playing facility at unbeatable prices, and expand the possibilities of using your Spectrum in yet another direction.

Alison Maguire
Applications Software Manager

SOFTWARE UPDATE

The latest cassette software for ZX® Computers

Chess Tutor 1

For 48K RAM Spectrum. **£9.95.**

Chess Tutor is a new way of learning all about chess – using your ZX Spectrum.®

It starts from the beginning by teaching you about the chess pieces and the way they move – including castling, en passant, promotion, check, checkmate, stalemate and perpetual check.

Then it teaches you the basic tactics – pins, forks, double attacks and skewers.

There are over 120 exercises and over 200 questions for you to answer – with demonstrations and hints from your ZX Spectrum when you want them.

You can choose which parts of the course you want – and even experienced players may be surprised at what they can learn from Chess Tutor.

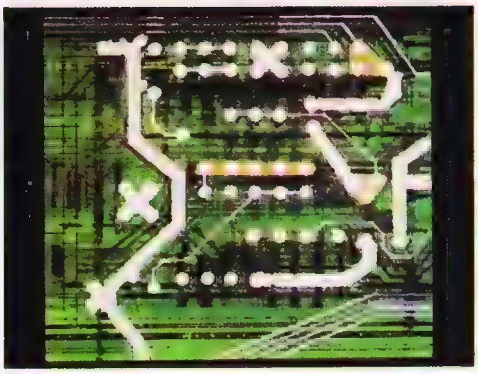
Musicmaster

For 48K RAM Spectrum. **£9.95.**

Musicmaster turns your ZX Spectrum into a musical instrument which will not only play tunes, but will also demonstrate key signatures, durations of notes, and scales.

You can write your own tunes – in any key – play them over and over again, save them on tape, modify them.

You can either write your music on a stave, or place a simple overlay on your Spectrum for a 17-note keyboard.



Make-a-Chip

For 48K RAM Spectrum. **£9.95.**

Make-a-Chip teaches you the basic elements of circuit design, shows you how they fit together, and then lets you design and test your own circuits.

When you have designed a circuit, you can give it inputs and outputs and your ZX Spectrum will check it for you. Then it will run it, or tell you what's wrong so that you can modify it.

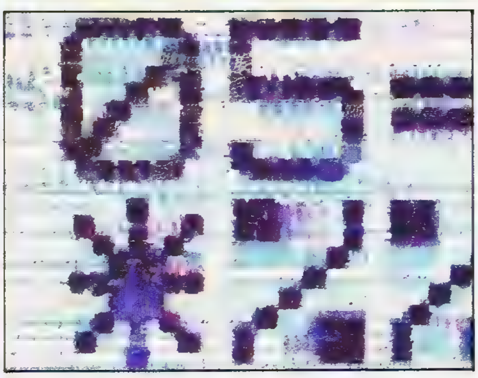
Make-a-Chip is a fascinating way of finding out how computer logic works.

Print Utilities

For 16K and 48K RAM Spectrum. **£9.95.**

Increase the printing and display facilities of your ZX Spectrum with the Print Utilities program.

Print Utilities enables you to enhance your programs by generating characters of eight different sizes which you can place anywhere on your screen.



Beyond BASIC

For 48K RAM Spectrum. **£9.95.**

Takes the agony out of assembler. Takes the mystery out of machine code.

Beyond BASIC gives you a deeper insight into the workings of your ZX Spectrum. It explains what happens inside your micro when you run a program, and it teaches you simple Z80 machine code programming.

A major feature of Beyond BASIC is that it enables you to write your own Z80 assembler programs – then you can actually see on your screen how they affect the ZX Spectrum memory and registers.

ZX INTERFACE 2®

The New ROM Cartridge/Joystick Interface

Loads programs instantly! Takes two joysticks! Just plug-in and play!

The ZX Interface 2 is the latest new peripheral for the ZX Spectrum® system. It enables you to use new ZX® ROM cartridge software: plug-in programs that load instantly. It allows you to use two standard joysticks, without the need for separate, special interfaces.

To use new ZX ROM cartridge programs, just connect Interface 2 to the rear of your Spectrum or Interface 1 and plug in the cartridge of your choice. The program is then loaded, ready to run!

You can use any joystick that has a 9-way D plug. Use one or two of them for extra fun with ZX ROM cartridge or Sinclair cassette programs – or with dozens of other Spectrum-compatible programs!

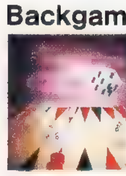
...AND BRAND NEW ROM CARTRIDGE SOFTWARE!

There's already plenty of choice of ZX ROM cartridge programs for your Spectrum. Some are old favourites, in an exciting new form. Others are new.

And now, thanks to ROM cartridge technology, you can run them *all* on a 16K RAM Spectrum, even if they were originally written only for 48K machines!

Every ROM cartridge program loads fast and faultlessly. No wires, no waiting, no worries about loading errors! All of them are affordably priced too, at £14.95.

Backgammon



Everything you need to play the famous and deceptively simple board game. Board, stones, rolling dice and doubling dice are shown in full colour and high resolution. Choose from four levels of skill to suit experts and beginners alike – full rules are included.

Space Raiders

Your skill is all that's stopping successive waves of aliens from destroying Earth. Use your gun base to attack. Shelter behind buildings... move out and blast the passing alien soaceship!

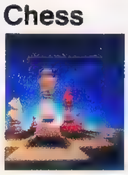
Full-colour high-resolution graphics with sound.

New! PSSST

Robbie the Robot sits in his garden. Help him fetch compost to cultivate his prize Thyrgodian Megga Chrysanthodil. Help him make the right choice of pesticide, to ward off devilish insects. Stop the insects breeding to overwhelming numbers before Robbie's plant has bloomed. PSSST is horticulture with a horrendous twist!

One and two player option, with a host of features including sound effects.

Chess



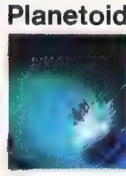
This sophisticated program does everything you'd expect at board game level, and much more besides.

The high-resolution chessboard and pieces are arranged in a row and column system, so it's easy to key in your moves.

At any stage of the game you can request the computer to suggest a move, reverse roles or change the level of skill.

Full-colour high-resolution graphics.

Planetoids



Dodge and swerve using your thrust button, turn on a planetoid... fire! But beware – the alien ship moves fast to destroy you with cluster bombs. And when it comes to the crunch, use your hyperspace button!

Full-colour high-resolution graphics with sound.

Hungry Horace

Horace is forever being chased around the park by guards.

He steals their lunch, eats pathway flowers and creates chaos in the park by ringing the alarm!

You'll have to be quick to keep Horace out of trouble!

Full-colour high-resolution graphics with sound

New! Tranz Am

Set in a future time ruled by cars and trophies, in a land where petrol replaces gold, and status is possession of the 8 Great Cups of Ultimate.

Driving your Super Blown Red Racer, use your skill to outwit and crash the Deadly Black Turbos. Use your instruments to locate and collect the trophies – before you overheat or run out of fuel.

A program with outstanding multi-directional movement, graphic features, and a playing area equivalent to more than 600 times actual screen area.

Horace and the Spiders

Guide Horace on the hazardous journey to the cobwebbed house full of poisonous spiders.

Safely in the house, you must move along cobwebs, choose a spot... and jump on it! The spiders will be in a frenzy – scuttling to repair their precious web.

And when a spider is spinning a new section, you're safe to attack and destroy it!

Kill all the spiders, and a new web appears... with even more spiders to catch.

Full-colour high-resolution graphics.

New! Cookie

You're Charlie the Chef, who keeps his ingredients locked in the larder. But if the ingredients escape, they bring the inedible Nasties with them!

You must daze the escaping ingredients with flour bombs, and knock them into the mixing bowl. Stop them getting into the dustbin, at all costs! And beware of Nasties that get into the mixing bowl!

Cookie is fast-moving panic in the pantry, with a cast of real characters. A program to make you smile – and sweat!

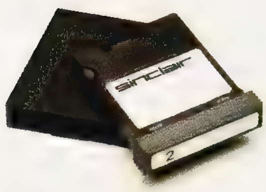
New! Jet Pac

As Chief Test Pilot of the Acme Interstellar Transport Company, your task is to deliver and assemble spaceship kits. On your way round the galaxy, you're free to collect precious stones and gold.

The catch? Rocket fuel is precious and scarce. And the aliens don't take kindly to the theft of their valuables. You'll need your wits and your lasers!

With a host of features, including multi-directional movement, explosions, sound effects and one and two player option.

ZX MICRODRIVE



NOW ON RELEASE

The ZX Microdrive[®] System – as you'd expect from Sinclair – is unique to the world of computing. It's a compact, expandable add-on system which provides high-speed access to massive data storage. With just one Microdrive alone (and Interface 1), you'll have at least 85K bytes of storage, the ability to LOAD and SAVE in mere seconds, the beginnings of a local area network of up to 64 Spectrums, *and* a built-in RS232 interface! The cost? Less than £50 for each Microdrive.

How to get ZX Microdrive

Spectrum owners who bought direct from us, by mail order, have been sent full details. Order forms are being mailed in strict rotation, so if you haven't yet received your order form please bear with us. We're making good progress in meeting the huge demand.

If you didn't buy your Spectrum by mail order, don't worry. Send us the form from the bottom of this page. We'll add your name to the mailing list, and send you details by return.

Each Microdrive costs £49.95. Interface 1 costs £49.95, but just £29.95 if purchased with a ZX Microdrive. Extra ZX Microdrive cartridges: £4.95.

How to order

Simply fill in the relevant sections on the order form below. Note that there is no postage or packing to pay on some purchases. Orders may be sent FREEPOST (no stamp needed). Credit card holders may order by phone, calling 01-200 0200, 24 hours a day. 14-day money-back option, of course. Please allow 28 days for delivery.

[®] ZX, ZX Spectrum, ZX Interface and ZX Microdrive are all registered trade marks of Sinclair Research Ltd.

sinclair®

Sinclair Research Ltd, Stanhope Road, Camberley, Surrey, GU15 3PS. Telephone: (0276) 685311.

To: Sinclair Research Ltd, FREEPOST, Camberley, Surrey, GU15 3BR. **ORDER FORM**

Section A: Hardware

Qty	Item	Code	Item Price £	Total £
	ZX Interface 2	8501	19.95	
	ZX Spectrum – 48K	3000	129.95	
	ZX Spectrum – 16K	3002	99.95	
	Postage and packing: orders under £90	0028	2.95	
	orders over £90	0029	4.95	
			TOTAL £	

Section B: Software **ROM CARTRIDGE PROGRAMS**

	G12/R	Planetoids	5302	14.95
	G9/R	Space Raiders	5300	14.95
	G13/R	Hungry Horace	5303	14.95
	G24/R	Horace and the Spiders	5305	14.95
	G28/R	PSSST	5307	14.95
	G30/R	Cookie	5309	14.95
	G29/R	Tranz Am	5308	14.95
	G27/R	Jet Pac	5306	14.95
	G22/R	Backgammon	5304	14.95
	G10/R	Chess	5301	14.95

CASSETTE PROGRAMS for ZX Spectrum

E9/S	Chess Tutor 1	4308	9.95	
E7/S	Musicmaster	4306	9.95	
E8/S	Beyond BASIC	4307	9.95	
E6/S	Make-a-Chip	4305	9.95	
L5/S	Print Utilities	4404	9.95	
			TOTAL £	

*I enclose a cheque/postal order made payable to Sinclair Research Ltd for £ ______

*Please charge to my Access/Barclaycard/Trustcard account no: ______

*Delete/complete as applicable.

Signature ______

Mr/Mrs/Miss ______

Address ______

COT 911

(Please print)

ZX Microdrive information request

Please add my name to the Microdrive Mailing List, *and* send me a colour brochure with full specifications of ZX Microdrive/Interface 1 ☐ *(tick here)*. You can use the above form to send us your name and address.

INDEX '83

Once again we roll around to the December issue, and it's time to plug in the word processor and alphasort an index for you.

GENERAL FEATURES

Adventures Part 1	Nov	49
Adventures Part 2	Dec	45
Beating The RS232 Blues	June	40
Cheap Print	Jan	74
Computer-generated Music	June	26
Curve Fitting	Dec	67
FELIX Knowledge Shops	Sep	18
Into Atari's BASIC Part 1	Mar	49
Into Atari's BASIC Part 2	May	20
Market Survey (16-bit machines)	July	63
Market Survey (micros)	May	75
Micro Interruptions Part 1	June	20
Micro Interruptions Part 2	Aug	21
PEEK And POKE Explained Part 1	Jan	33
PEEK And POKE Explained Part 2	Feb	74
PEEKING The Spectrum	Mar	45
Second Helping: The BBC Computer programme	Feb	31
Sharp Speed-up	Aug	28
Software Protection	Sep	31
Sweet Sixteen (16-bit micros)	July	28
Terminology Translated	Mar	66

REGULAR FEATURES

Club Call	Feb	92
	Mar	72
	Apr	85
	May	86
	Sep	78
	Oct	70
	Nov	96
# FILE		
Silicon Valley Visit	Jan	65
The Hobbit/Star Maze	Mar	41
IBM versus Apple	Apr	44
Welsh BASIC and reference books	June	49
BOOK PAGE		
Beginner's books	Jan	80
Alternative language books	Mar	76
Books for the ZX Spectrum	May	24
Educational Books	Nov	45
Books on the CBM64, IBM, CP/M	Dec	80
PROCopinion		
Darkstar	Sep	62
Electron launch and Acorn User Show	Nov	88
VIEWPOINT		
Computer Literacy	Jan	27

SERIALISED ARTICLES

Elegant Programming	Jan	57
	Feb	45
	Mar	28
	Apr	81
Getting More From The 64	Dec	29
Going FORTH Again	May	45
	July	37
Languages In Use: Introduction	Feb	20
GINO-F	July	44
COMAL	Sep	21
LISP	Oct	25
Learning FORTH	Nov	68
	Dec	52
Pascal Profile	Jan	20
	Feb	52
	Mar	56
	Apr	64
Problem Page	Aug	46
	Sep	28
	Oct	62
	Nov	81
	Dec	76

HARDWARE REVIEWS

Apple III	Jan	40
Colour Genie	Mar	21
Epson HX-20	Mar	33
Fujitsu FM-8	Feb	37
Gemini Galaxy 1	Apr	50
Ikon Hobbit Tape Drive	July	55
Jupiter Ace	May	53
Laser 200	Nov	37
Lynx	June	57
Olivetti Praxis 30 daisywheel	July	19
Oric-1	May	37
Premier Publications Dragon Disc Drive	Aug	31
Sharp MZ-3541	Oct	49
Sharp MZ-700	Oct	41
Sharp PC-1251 hand-held	Apr	33
Smith-Corona TP-1 Printer	Mar	62
Sord M5	Aug	36
Tandy Model 4/MC10/Model 100	Sep	65
Torch Z80 Disc Pack	Sep	41
ZX Lprint Centronics Interface	Nov	25

Sharp's MZ-700 October p41.

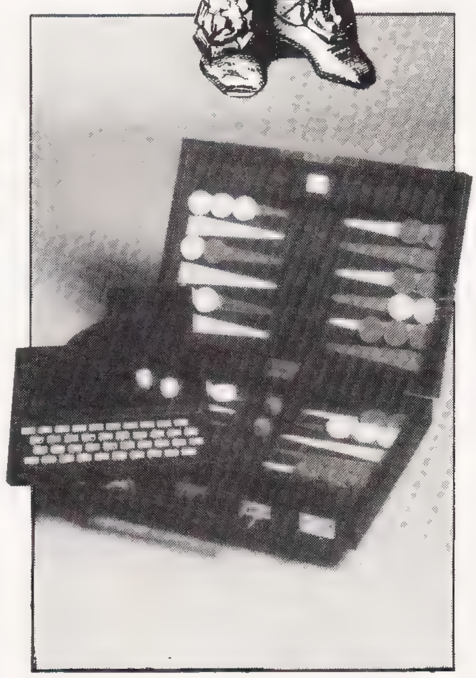
Pascal profile, January-April.

SOFTWARE REVIEWS

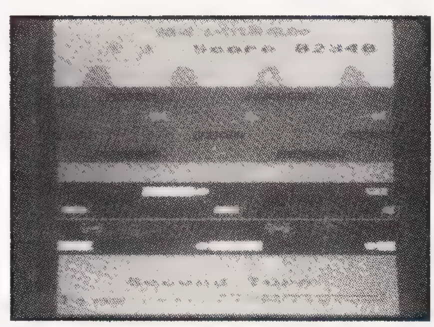
Adventurous Software	Oct	86
Arrow Tape Operating System	Dec	64
Getting Adventurous 1	Apr	58
Getting Adventurous 2	May	32
Getting Adventurous 3	June	63
Mikro Magic	Dec	18
Three FORTHs	Nov	76

SOFTWARE

A Better TRSDOS	TRSDOS Ver 2. 3/BASIC	Oct	65
A Micro Data Base 1	TRSDOS	Apr	73
A Micro Data Base 2	TRSDOS	May	61
Atari Renumber	Atari BASIC	July	17
BBC String Store	BBC BASIC	Aug	48
BBC Touch Typing Tutor	BBC BASIC	Dec	71
BBC Word Processor	BBC BASIC	Nov	18
Backgammon	ZX81 BASIC	Sep	48
Bombproof Tandy	TRS-80 BASIC	July	32
Character Generator	Dragon 32 BASIC	Sep	57
Chilotux	TRS-80/Z80 Assembler	Feb	84
Dragon Graphicks	Dragon 32 BASIC	Dec	22
Flashing Microtan Cursor	6502 assembler	Oct	32
Floating Point Dictionary	Jupiter Ace FORTH	Nov	84
Froglet	BBC BASIC	Apr	25
Froglet Update	BBC BASIC	June	67
Genie Space Saver	Genie/Z80 Assembler	July	23
Improved VIC Editor	VIC-20 BASIC	Oct	81
Indexer	TRS-80 BASIC	June	69
Kepler's Revenge	BBC BASIC	Jan	88
MCBAS Converter	Genie/assembler	Dec	38
NASCOM Find	Z80 Assembler	Feb	66
NASPEN Append	NASCOM/Z80 assembler	Apr	20
Planetfall	Sharp BASIC SP-5025	Aug	52
Rescuing Source Code	EDTASM/Z80 Assembler	Nov	28
Slingshot	BASIC/GRAPHPAC	Oct	19
Tailoring VIC's Characters	VIC-20 BASIC	Feb	60
Tape Append	PET BASIC	Nov	92
Tube Cube	Spectrum BASIC	June	34
VIC Blow-up	VIC-20 BASIC	Jan	70
VIC Format Converter	VIC-20 BASIC	July	52
Valley Variations	TRS-80/BBC/Spectrum	Apr	38
Z80 Disassembler	BASIC	Oct	57



ZX81 Backgammon September p48.



Froglet April p25.


THE FABULOUS
CASSETTE
50
FROM cascade
50 GAMES ON ONE GREAT CASSETTE
ONLY
£9-95
DON'T MISS THIS
INCREDIBLE OFFER
50 FANTASTIC
GAMES ON
ONE CASSETTE
ONLY £9.95 (INC.P&P and VAT)
SPECTRUM
50 fabulous games
VIC 20
50 fabulous games
EXPRESS DELIVERY - ORDER NOW
Please send me by return of post, Cassette 50 at £9.95 per tape. I enclose a cheque/postal order for £ made payable to Cascade Games.
Name
Address
Post Code
Country
Dealers & Stockists enquiries welcome.
Please send me (tick appropriate box)
Spectrum
VIC
cascade
Cascade Games,
Suite 4, 1-3 Haywra Crescent,
Harrogate, N. Yorkshire, HG1 5BG.
England.
Telephone: (0423) 504526
CT/12

Peter Freebury

A BOLT FOR THE COMMODORE

Several utility cartridges for the Commodore 64 are now appearing, and in search of a faster tape load, our reviewer loosed an ARROW at his computer.

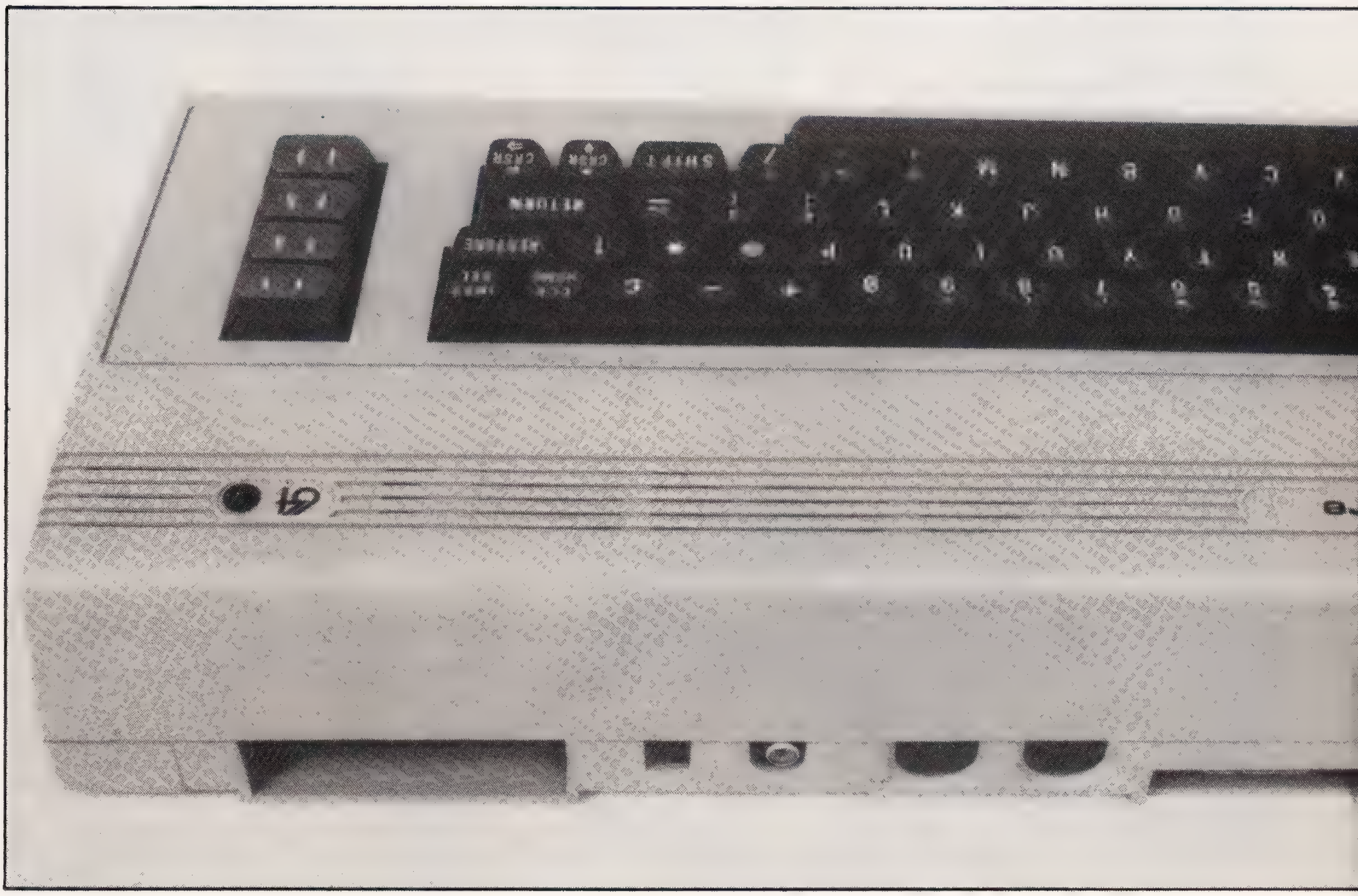
ARROW is not a new name to be associated with Commodore computers; some years ago there was a similar-sounding device. That too, was marketed in this country by Supersoft. Both devices offer the ability to SAVE, LOAD, APPEND and VERIFY on cassette in a very much shorter time than normal. The cassette continues to turn at normal speed but the computer writes/reads information at a much faster rate. This latest offering is in the form of a ROM cartridge and simply plugs into the cartridge slot at the rear of the Commodore 64.

Although the price (£39.00) may make some of the faint-hearted think twice, I would certainly recommend that they think yet again! The present Commodore 64 ARROW is as fast to SAVE or LOAD a program as the 1541 disc drive, 21K of program taking about 50 seconds as opposed to 430 seconds using normal LOAD/SAVE commands. This is an absolute boon to the enthusiastic programmer. Or to the paranoic programmer who firmly believes that there is going to be a mains failure at any moment and wants to SAVE his latest creation every 10 minutes!

The cost of ARROW is about one-sixth of the cost of a disc drive so I suppose it is worth considering on that score alone, but there is more to ARROW than a rapid cassette system. Within its plastic exterior lie some very acceptable jewels indeed! The concept of 'toolkits' seemed to start in this country with those for the Commodore PETs. A number of similar utilities have appeared since then, the most noticeable ones being for the Sinclair Spectrum and the Commodore VIC20. ARROW now provides you with a few of the most useful routines a busy programmer is likely to need.

NOT ONLY BUT ALSO

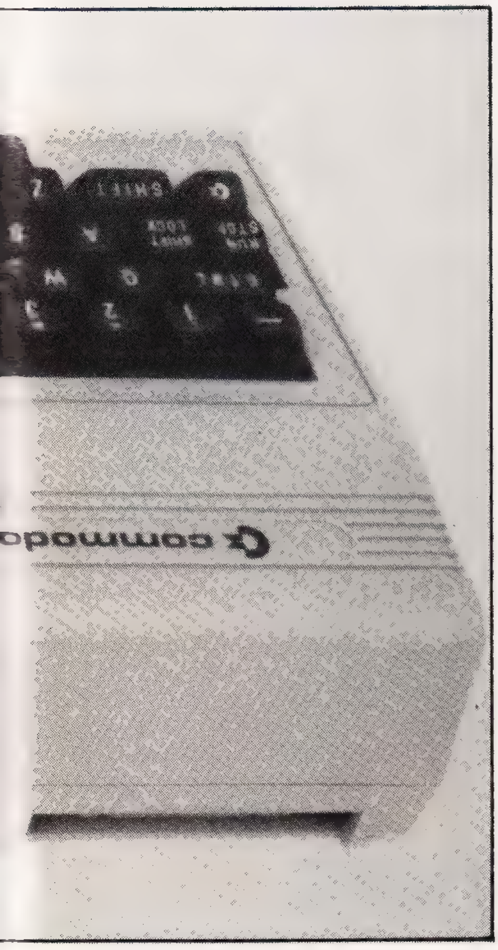
Let us work through ARROW command by command. First, as we have already said, there is the ability to fast SAVE, LOAD, VERIFY and APPEND with or without a file name (maximum 14 characters). You may only utilise this rapid LOAD with programs previously SAVEd using ARROW, a minor point but one that we should perhaps note. The commands for the above are <S, <L <V and <A. There is a variant to <S that shows just what thought has gone into the production of ARROW. As you are no doubt aware, cassette tapes come with variable length leaders; <S gives you four seconds 'lead in' before starting to SAVE to the tape, but if in doubt then use <T which will give a 10 second lead time — so simple but oh so safe! As your programs will now take up much less tape you will be able to get more programs on the standard C20, C30 etc.



Will you have to look at the tape counter like a hawk as you wind on to the nth program? No, ARROW will do this for you. Simply type <Pn (n = 1 to 9) and you will be told to PRESS FAST FORWARD on the tape deck. ARROW will stop the tape at the chosen spot and you can <S or <L as required. Each 'block' of tape is long enough for 16K of program. In addition, after any <S or <L etc, the screen displays the number of bytes used. That can be quite an eye-opener on occasion!

The programming aids hinted at earlier are few but extremely useful. The first is an AUTO numbering facility which takes the form <N x,y where x is the first line number required and y is the increment. If either x or y are omitted then a default of 10 is assumed. Typing <Q breaks you out of Auto Numbering. Should you not need one of the line numbers generated then just press return and ARROW will skip to the next number in sequence. Although this sounds obvious it is very useful when you are copying someone else's program that has not been structured as neatly as your own would be!

DELETEing blocks of program lines is achieved by <D x,y where x is the first line and y is the last line to be deleted. Both x and y *must* be specified but do not have to be actual program lines. So <D 1,63999 would delete your entire program regardless of whether you actually have lines numbered 1 or 63999.



FINDing a character or group of characters is to my mind one of the most useful utilities available to the programmer. The form is <F "abcde. . . ." where abcde etc are the characters you wish to find. They may be any set of characters that you can type from the keyboard between quotes. Sometimes you will have to think carefully of how to specify the required search. <F "A" to find the use of variable 'A' may result in more than you bargained for but <F "A=",<F "=A" or <F "+A", for example, will narrow the search down considerably. <F finds the occurrence of a set of characters in or out of quotes equally easily so <F "REM" will find all your REMs as well as "THE REMAINDER IS. . .". The LISTing of the lines containing your specified characters may be slowed down by using the CTRL key just as in a normal LISTing.

In addition to FIND you may FIND and REPLACE using <F "yyyy"/zzzzz: this will find the occurrence of the search string and substitute the replacement string. Both versions of the line are displayed on the screen. If you wish to cancel the change merely cursor up to the original version and press return. To continue further replacement type < and press return.

ARROW also has a neat hexadecimal calculator built in. <H will display two counters (initially zero) at the bottom of the screen. Enter a hexadecimal number of up to four digits and it will appear in the right hand counter — the decimal equivalent will appear on the left. Keying '*' will reverse the process and decimal numbers keyed in will appear on the left and their hexadecimal equivalents on the right. To add or subtract numbers use the plus, minus or equals keys as you would using an ordinary calculator. To exit back to normal use type 'X'.

DELVING DEEPER

The Commodore 64, unlike its ancestors the PETs, does not have a built-in machine language monitor. Yes, you've guessed it — ARROW does! Type <X to enter the monitor: this will initially display the contents of the 6510 processor's main registers when the monitor was entered. You may display these registers at any time by typing 'R'. The registers displayed are the Program Counter, Status Register, Accumulator, X Register, Y Register and Stack Pointer.

Other commands available in the monitor are as follows:

.M xxxx yyyy (where xxxx and yyyy are addresses in Hex) will display the contents of the block of memory between these two addresses, eight bytes to a line. The contents of the memory may be changed by typing over the existing Hex values followed by return.

.S "filename",01,xxxx,yyyy will SAVE the memory between the two addresses in normal (non-ARROW) format. Note that 'yyyy' must be one greater than the actual end address of the block.

.L "filename" will LOAD a block of memory SAVEd with the .S command. If the filename is omitted then the first file found will be LOADed.

.G xxxx will commence execution of a machine language program at the specified address 'xxxx'. A break instruction ($00) in the program will cause a return to the monitor.

.H xxxx yyyy ab cd ef (etc) will search for the specified sequence of Hex bytes (ab cd ef) within the memory block (xxxx to yyyy) specified. You may search for a single byte or for as long a sequence as you can fit on one line.

.X exits from the monitor back to BASIC.

Finally there are some ARROW functions that will be mainly of interest to machine code programmers:

<M xxxx,yyyy,zzzz will move a block of memory (xxxx to yyyy) to a new start address 'zzzz'.

<C xxxx,yyyy,zzzz will compare two blocks of memory. The comparison is between the first block (xxxx to yyyy) with the second block starting at 'zzzz'. Any differences are displayed on the screen.

<S "filename",xxxx,yyyy is used to SAVE a block of memory at ARROW speed. That <T command may also be used with a specified start and end address. To LOAD or VERIFY a program SAVEd in this way you must shift the 'L' or 'V' which is equivalent to LOAD "filename",1,1 in normal BASIC.

IN CONCLUSION

Although ARROW might not be quite as versatile as some disc-based 'Programmer's Aids' or 'improved' BASICs, it is completely compatible with Commodore BASIC. It is a 'plug-in' ROM cartridge so no LOADing time is required and I leave it attached permanently. It has already saved me many, hours with the routines it contains. I would unhesitatingly recommend it to anyone.

The address of Supersoft is Winchester House, Canning Road, Wealdstone, Harrow, Middlesex HA3 7SJ (telephone: 01-861 1166).

Don Thomasson

CURVE-FITTING

Joining the dots is fun for kids but a matter of rather more seriousness for the mathematicians. Here's a procedure to calculate an equation which best fits a set of points.

0					
	0.5				
0.5		-0.134			
	0.366		-0.098		
0.866		-0.232		0.062	
	0.134		-0.035		0.01
1		-0.268		0.072	
	-0.134		0.036		
0.866		-0.232			
	-0.366				
0.5					

Fig. 1 Six-column difference table.

One of the less happy memories of my long-distant schooldays relates to the drawing of graphs. The lines had to be firm and fine, not 'painted', and had to link the given points in a convincing manner. My maths master often remained unconvinced, and I could have made good use of the program described here — except that no computers existed in those far-off days!

The program is based on two limiting assumptions: that the x co-ordinates of the points are equally spaced, and that the curve can be expressed by the equation;

$$y = C_0 + C_1*x + C_2*x^2 + C_3*x^3 + \ldots + C_n*x^n$$

where n is the number of given points. The difference between the successive X values will be called d.

If a 'difference table' of the kind illustrated in Fig. 1 is drawn up, the value of C_n can be determined by direct calculation. The left-hand column contains the successive y values for the given points, and each subsequent column contains the difference between successive values in the preceding column. The nth difference column contains a single entry. C_n is found by dividing this entry by n! (n factorial) and then dividing the result by d^n. The C_n term can then be removed from the equation, giving a revised set of y values from which a new difference table can be constructed.

The new table has $n-1$ columns, and the last column contains two entries, which should be the same. In case they are not, because of minor calculation errors, it is best to take their average, and dividing that by $(n-1)!d^{(n-1)}$ will give constant C_{n-1}. This process is continued until all the constants have been evaluated.

A simple case may help to make the method clear — see Fig. 2. The original difference table, (a), shows that $C_2 = 1/(2!*d^2)$. Taking d as unity, C2 = 0.5. Subtracting $0.5x^2$ from the left hand column, the difference table at (b) can be produced. The value of C_1 is seen to be 0.5/d, ie 0.5. Subtracting 0.5x from the left hand column gives the table at (c), and this shows $C_0 = 3$. The equation for the curve through the given points is therefore:

$$y = 3 + 0.5x + 0.5x^2$$

For larger numbers of points the process is tedious and error-prone, but it is quite suitable for computer use, subject to certain limitations.

THE PROGRAM

The program was written for a ZX Spectrum, but with the exception of the final plotting routine it should transfer almost directly to any other form of BASIC.

Subroutine 1000 deals with data input, beginning with NUM, the number of given points, which is used to dimension the arrays. The base value and increment of x are then set up, and finally the y values. For reasons that will be explained later, the x values are scaled so that the increment becomes 0.2. The y data is stored in A(1,N).

Since it is necessary to modify the y data as the program proceeds, it is next copied into A(2,N), which provides the data for the first column of the difference table. The main routine at line 2000 follows.

The loop starting at line 2000 calculates one C constant during each iteration, beginning with C_{NUM}. Subroutine 3000 sets up the number and length of each column in turn, and calls subroutine 4000 to calculate the difference table, the average values of the individual columns being stored in array B. Only one entry from this array is used in a given iteration, but it is easier to set up all the values, and they are useful in debugging and other checks.

Subroutine 5000 then displays at least part of the table. The Spectrum will show six columns, if a little confusion is accepted, but our old faithful, the Sorcerer, will show a good deal more, having twice the screen width. This routine is not essential, but it gives an insight into the process and the way it is getting on.

Lines 2030-2060 then calculate the factor by which the last column entry must be divided to find the value of the constant C. The factorial is generated in such a way that the value of the factor is kept within relatively small limits, and the need for this will be examined later.

The constant is calculated from B(K)/FAC, and the result is set in array C. Then lines 2080-2100 modify the values in array A(2,N), ready for the construction of a further difference table.

Finally, the average of column 0 is subtracted from the individual entries, the average being set as C_0. If the subtraction leaves non-zero values, the process has failed to some extent, but the errors may not be significant.

Subroutine 6000 plots the given points and the curve. It is particular to the Spectrum, and those who have no plotting facilities may prefer to display the given points and interpolated values side by side. However, the routine as given may suggest an approach for use with other

(a)			(b)			(c)		
3			3			3		
	1			0.5			0	
4		1	3.5		0	3		0
	2			0.5			0	
6			4			3		

Fig. 2 Difference table development

machines.

It is necessary to scale the display to fit the screen, and this is done first. The range of X values is related to the number of dots which can be displayed in the screen width, yielding XSCALE. The range of Y values (in the given data) is then determined via MAX and MIN, and related to the number of dots in the screen height with an allowance for overspill, since the curve may exceed the range of the given points. The given data is then plotted in the form of small crosses, using the given x values and the y values from A(1,N), which have been left unaltered.

The actual curve plot begins at 6160, one dot being plotted for each of the possible columns in the screen width, less a few to ensure no overspill. Where the y value strays outside the screen height, plotting is suspended.

LIMITATIONS

The program works well enough for small numbers of given points, but may give peculiar results with more than 10 points or so. The reason is that some very large numbers can be involved. The table of factorial values in Fig. 3 underlines this, especially when it is compared with the accuracy limits of various form of floating point notation.

n	n!
1	1
2	2
3	6
4	24
5	120
6	720
7	5040
8	40320
9	362880
10	3628800
11	39916800
12	479001600
13	6227020800
14	87178291200
15	1307674368000
16	20922789888000
17	355687428096000
18	6402373705728000
19	121645100408832000
20	2432902008176640000

Fig. 3 Table of factorials.

Single precision floating point uses 24 bits in the mantissa, which can therefore represent 16,777,216 different values, though this number is sometimes halved by an

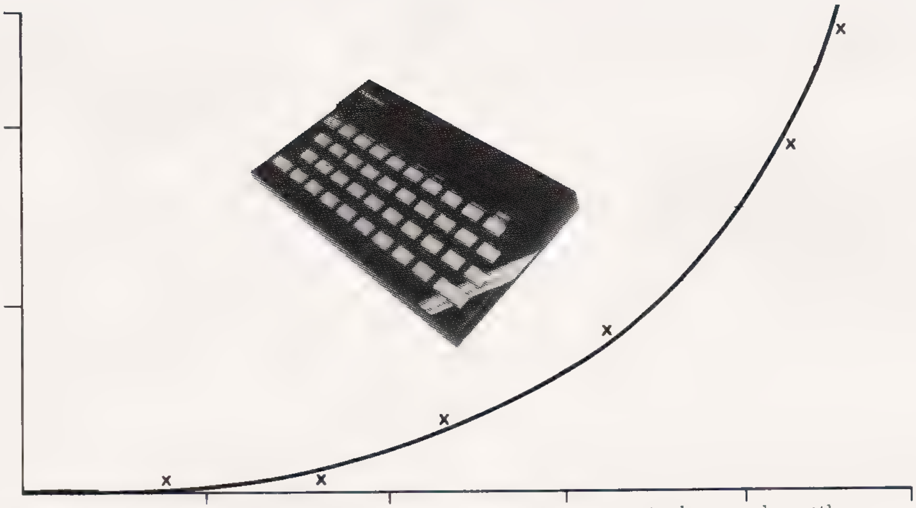
```
100 GO SUB 1000
110 FOR X=1 TO NUM
120 LET A(2,X)=A(1,X)
130 NEXT X
140 GO SUB 2000
150 GO SUB 6000
160 STOP
1000 INPUT "NUMBER OF POINTS ? ";NUM
1010 DIM A(NUM+1,NUM): DIM B(NUM): DIM C(NUM)
1020 INPUT "BASE VALUE OF X ? ";BASE
1030 INPUT "INCREMENT OF X ? ";INCR
1040 LET M=0.2/INCR
1050 LET MBASE=BASE*M
1060 FOR N=1 TO NUM
1070 PRINT AT 20,0;"INPUT Y VALUE FOR X = ";BASE+(N-1)*INCR
1080 INPUT A(1,N)
1090 NEXT N
1100 CLS
1110 RETURN
2000 FOR K=NUM TO 1 STEP -1
2010 GO SUB 3000
2020 GO SUB 5000
2030 LET FAC=5/K
2040 FOR J=1 TO K
2050 LET FAC=FAC*J*0.2
2060 NEXT J
2070 LET C(K)=B(K)/FAC
2080 FOR Z=1 TO NUM
2090 LET A(2,Z)=A(2,Z)-C(K)*((MBASE+(Z-1)*0.2)↑(K-1))
2100 NEXT Z
2110 NEXT K
2120 LET TOT=0
2130 FOR J=1 TO NUM
2140 LET TOT=TOT+A(2,J)
2150 NEXT J
2160 LET C(1)=TOT/NUM
2170 FOR J=1 TO NUM
2180 LET A(2,J)=A(2,J)-C(1)
2190 NEXT J
2200 GO SUB 5000
2210 FOR K=1 TO NUM
2220 LET C(K)=C(K)*(M↑(K-1))
2230 NEXT K
2240 RETURN
3000 FOR J=2 TO NUM
3010 LET COL=J
3020 LET LIM=NUM-J+1
3030 GO SUB 4000
3040 NEXT J
3050 RETURN
4000 LET F=0
4010 FOR X=1 TO LIM
4020 LET H=A(COL,X+1)-A(COL,X)
4030 LET F=F+H
4040 LET A(COL+1,X)=H
4050 NEXT X
4060 LET B(COL)=F/LIM
4070 RETURN
5000 CLS
5010 FOR Z=1 TO NUM
5020 FOR Y=1 TO NUM
5030 IF Y+Z>10 OR Y+Z>NUM+1 OR Y>6 THEN GO TO 5050
5040 PRINT AT 2*(Z-1)+Y 5.5*(Y-1);A((Y+1),Z)
5050 NEXT Y
5060 NEXT Z
5070 PRINT AT 0,20;K
5080 RETURN
6000 CLS
6010 LET XSCALE=253/(NUM*INCR)
6020 LET MIN=0: LET MAX=0
6030 FOR K=1 TO NUM
6040 IF A(1,K)>MAX THEN LET MAX=A(1,K)
6050 IF A(1,K)<MIN THEN LET MIN=A(1,K)
6060 NEXT K
6070 LET YSCALE=160/(MAX-MIN)
6080 FOR K=1 TO NUM
6090 LET X=((K-1)*INCR)*XSCALE
6100 LET Y=(A(1,K)-MIN)*YSCALE
6110 PLOT X,Y+10
6120 DRAW 2,0
6130 PLOT X+1,Y+9
6140 DRAW 0,2
6150 NEXT K
6160 FOR K=1 TO 255
6170 LET X=(K-1)/XSCALE+BASE
6180 LET Y=C(NUM)
6190 FOR J=NUM-1 TO 1 STEP -1
6200 LET Y=Y*X+C(J)
6210 NEXT J
6220 LET L=(Y-MIN)*YSCALE+10
6230 IF L>0 AND L<175 THEN PLOT K,L
6240 NEXT K
6250 RETURN
```

Listing 1. Curve fitter for the Spectrum.

automatic rounding process. If an attempt is made to calculate 11! = 39,916,800, an accurate result may or may not be obtained, depending on the exact working of the floating point system. The exponent will have a value of 26, and $2^{26} = 67,108,864$. The mantissa will have a value of (nominally) 0.5948066. This is possible, the hex form being 984540. However, an error of one bit, perhaps induced by rounding, will give 11! as 39,916,804, and though that may seem a negligible error, only one part in ten million, it can be enough to cause chaos, because the program deals with small differences that are multiplied by high powers of X.

This particular problem is avoided in the program by calculating the factorials in such a way that the dividing factor remains within comparatively small limits, but while that will give accurate values for C_n there may be errors in $C_n \star X^n$ which make subsequent tables very inaccurate.

These points have been made as a warning that the results obtained with too many points may be disappointing. In a particular case, with the same set of points in use, the first eight have a perfect result, the first 10 matched the



points, but in a slightly odd way, the first 12 missed one point completely, and the first 13 produced nonsense.

Some extension of the working range can be obtained by using double-precision, if it is available, and with some computers it will be found that x ^ n is significantly less accurate than x multiplied by itself n times. The power calculation uses logs, and may well be faster, whereas the multiplication is direct, more accurate, but slower. If multiplication is preferred, a function or procedure may be used to execute the task, this allowing comparisons to be made with minimum change.

This excursion into the finer points of mathematical calculation may be relevant to other programs. It is only too easy to fall into the trap of assuming that calculations are unconditionally accurate, whereas they may go seriously astray, especially in respect of small differences between large numbers.

However, in this instance, the problems only arise with a large number of points, and the program should be convincing enough, even for a cynical maths master . . .

James Tyler

BBC TOUCH TYPING TUTOR

The typesetters upstairs never cease to amaze me. They can type without watching where their fingers are! Amazing! Now you can learn to do it too.

As the title suggests, the aim of this program is to try and help you improve your typing tactics and look a bit 'cooler at the keyboard'. I was inspired into writing it when I delved into a vintage September 1980 edition of *Computing Today*, and stumbled upon a smaller program, written for an Ohio Superboard which had similar intentions.

The Touch Typing Tutor starts off by drawing an accurate representation (except for the cursor keys which I've omitted because they wouldn't all fit on the screen) of the Beeb's keyboard in MODE 1 graphics. The idea is that the user then uses this picture when trying (!) to press the correct keys, instead of looking down at the actual keyboard — easer said than done!

The program attempts to teach you touch typing by flashing the required key in its position on the screen and then waiting for an input. If the correct key is pressed, the equivalent one on the screen simply stops flashing. If, however, an incorrect key is pressed, then the one that should have been pressed will continue to flash and the one that was pressed will blank out.

Using this technique, it is hoped that the user (or pupil) will gradually learn the relative positions of each of the keys by touch alone.

The program is split into two separate parts called, surprisingly, Part 1 and Part 2, either of which may be selected using the menu on the title page.

Part 1 merely tests the user by prompting him or her to press a randomly chosen key. This, of course, uses the method outlined above. Whenever the wrong key is pressed, a counter (MC%) is incremented and the ASCII value of the key that was supposed to have been pressed is stored in an array — M%(MC%).

When the program has tested the user for 10 different keys, it looks to see how many mistakes were made. If more than 20% of the entries were wrong, then the program tests the user on the keys that he or she mistyped by choosing values at random from the array M%. In this way, the user may practise on the keys that they are particularly fond of missing (all of them?).

So, in fact, Part 1 may be running at any time in one of two possible modes — the initial 'random' mode or 'repeat' mode. This is indicated by the value of the repeat flag (RF%) where RF% = 1 means repeat mode, while RF% = 0 means random mode. The number of tests done in repeat mode is proportional to the number of mistakes that were made in random mode, ie MC% *2.

Pressing ESCAPE at any time will return you to the title page.

THE FAMOUS PART 2

Part 2 of the Touch Typing Tutor lets you get a bit more adventurous and practise typing in whole lines of BASIC. A line of text is printed ▶

on the screen below the keyboard diagram and a flashing cursor sits below that inviting you to enter the line yourself. You need not use the DELETE key since the program will only accept an input if it corresponds to the next character in the BASIC line (it is worth noting at this point that you ***cannot*** cheat and use the COPY key!). If you press the wrong key the program will respond in the usual tradition and flash the key that you should have pressed. So there are still no excuses for looking down at the real keyboard.

If, while entering a line of BASIC text, you are suddenly confronted with the question "CAPS LOCK CORRECT?" it means that you are pressing the correct key but have obtained the wrong symbol, $ instead of 4 for example. This sort of thing occurs when you have just entered some lower case text and are then trying to enter a BASIC keyword, only to find the CAPS LOCK is still off.

The additional exciting features of Part 2 only become apparent when you finish entering the line of text and press RETURN. You will then be told your typing accuracy for that line (100% indicates no mistakes) and your typing speed in characters per minute. This could probably be changed to words per minute if you decide on a value for the length of an average word, but it wouldn't be as accurate.

PROGRAM DETAILS

I wouldn't exactly say that this program was structured — there are a few too many GOTOs in it for a start — but you'll notice that I've tried to use procedures and REPEAT-UNTIL loops when I've felt the urge. Most of the procedures have fairly self-explanatory names (no prizes for guessing what PROCdraw keyboard does!) but a few are worthy of mention.

PROCKEY (CODE %, colour%) is probably the most important routine in the listing, and the operation of the whole program centres around it. Its purpose is to draw any key in the correct position on the screen, in the specified colour.

As an example, suppose we wanted to make the 'Q' key (ASCII code 81) flash, then the following would be done:
(i) First define colour 1 as flashing black/white (colour 15);
VDU 19, 1, 15 ; 0
(ii) Then call the procedure:
PROCKEY (81,1)
All this is done in MODE 1, of course.

It is this procedure that is indirectly responsible for all the heaps of data present in lines 620 to 660 of the listing. This data holds the information relating to each key on the keyboard: the ASCII codes for the shifted and unshifted symbols, X and Y co-ordinates for the screen positions, and the physical width of each key. Each line of data corresponds to a row on the keyboard.

PROCDECODEDATA has the exciting task of extracting each of the above parameters from the data provided.

PROCprintline is used by Part 2 of the Touch Typing Tutor to print a line of BASIC text on the screen. This is done by choosing, at random, a line from within the program itself and PEEKing it. With a little bit of help from the detokenising routine found in ROM at &B53A, the line is printed on the screen. All the line numbers are generated randomly to save time and effort (I admit it).

The machine code mnemonics found in the listing from lines 2300 to 2470 assemble to form a short routine which is called before and after the execution of PROCprintline.

This routine alters the Operating System Write Character (OSWRCH) vector to point at a new routine called STORELINE. This then means that whenever a character is sent to screen (in the circumstances, during PROCprintline) it does so via the STORELINE routine. All that STORELINE does is make a note of the character about to be printed on the screen by storing a copy of it elsewhere in memory. When this has been done, the character which is in the accumulator is sent on to OSWRCH as usual.

When PROCprintline has been executed, the OSWRCH vector is reset to its default value.

What's the purpose of it all anyway? Well, in Part 2 of the Typing Tutor the program has to know if the user is entering a correct line. This is quite easy now since there is a separate copy of it in memory, which was made by STORELINE previously. Variable 'line' holds the starting address of this copy. It is relatively simple to check if user's last input was valid by comparing it with the corresponding character stored in the address given by (line + horizontal position of the cursor) — 1. If you didn't do it that way, you might find yourself trying to compare characters stored in the screen memory, but since that's bit-mapped, it would get a bit slow and complicated!

Listing 1. BBC Touch Typing Tutor.

```
100 REM ******************************
110 REM *     Touch Typing Tutor     *
120 REM * JAMES TYLER April/May 1983 *
130 REM *  Runs in a 32K BBC Micro   *
140 REM ******************************
150 PROCinitialise
160 MODE1:VDU29,0;500;
170 COLOUR2:PRINTTAB(11,1)"Touch Typing Tutor"
180 PRINTTAB(11)STRING$(18,"_")
190 PROCdrawkeyboard
200 ON ERROR GOTO2510
210 REPEAT
220   VDU28,0,31,39,19
230   PROCMENU
240   CLS:COLOUR2
250   PRINTSPC8"Part ";P$(A-1)
260   VDU28,0,31,39,20,19,1,6;0;17,1
270   ON A GOTO280,290
280   PROCPART1
290   PROCPART2
300   UNTIL FALSE
310 DEFPROCKEY(CODE%,colour%)
320 IF CODE%=&7F CODE%=&FC
330 IF CODE%=&87 CODE%=&FD
340 RESTORE:REPEAT
350   PROCDECODEDATA
360   UNTIL CODE%=U% OR CODE%=L%
370 IF colour%=1 1%=L%:u%=U%
380 GCOL0,colour%
390 DEFPROCkey
400 Y%=(ASC(Y$)-48)*80
410 IF LEN(Y$)>1 W=VAL(MID$(Y$,2,3)) ELSE W=1
420 MOVEX%*10,Y%:PLOT1,S%*W,0:PLOT1,C,-C
430 PLOT1,0,-S%+1.5*C:PLOT1,-C,-C
440 PLOT1,-S%*W,0:PLOT1,-C,C
450 PLOT1,0,S%-1.5*C:PLOT1,C,C
460 IF I%=0 ENDPROC
470 MOVEX%*10+S%/3,Y%-5:VDUU%
480 IF U%>63 AND U%<91 ENDPROC
490 PLOT0,-32,-32:VDUL%
500 ENDPROC
510 DEFPROCdrawkeyboard
520 LOCALI%
530 GCOL0,3:VDU5
540 RESTORE:FOR I%=1 TO 70
550   PROCDECODEDATA:PROCkey:NEXT
560 MOVE1132,136:PRINT"RETN"
570 VDU4
580 ENDPROC
590 DEFPROCDECODEDATA:LOCALK%
600 READK$,X%,Y$:K%=EVAL("&"+K$)
610 U%=K% DIV256:L%=K% AND&FF:ENDPROC
620 DATA2020,28,09,2020,1,1.75,2020,8,11.5,5A7A,20,1,58
78,28,1,4363,36,1,5676,44,1,4262,52,1,4E6E,60,1,4D6D,68,1
,3C2C,76,1,3E2E,84,1,3F2F,92,1,2020,100,11.5,FC20,112,1,F
D20,120,1
630 DATA2020,1,2.75,2020,8,2,4161,16,2,5373,24,2,4464,3
2,2,4666,40,2,4767,48,2,4868,56,2,4A6A,64,2,4B6B,72,2,4C6
C,80,2,2B3B,88,2,2A3A,96,2,7D5D,104,2,0D0D,112,22
640 DATA0909,2,31.5,5171,14,3,5777,22,3,4565,30,3,5272,
38,3,5474,46,3,5979,54,3,5575,62,3,4969,70,3,4F6F,78,3,50
70,86,3,4040,94,3,7B5B,102,3,605F,110,3
```

```
 650 DATA2020,2,4,2131,10,4,2232,18,4,2333,26,4,2434,34,
4,2535,42,4,2636,50,4,2737,58,4,2838,66,4,2939,74,4,3020,
82,4,3D2D,90,4,7E5E,98,4,7C5C,107,4
 660 DATAF220,20,5,F320,28,5,F420,36,5,F520,44,5,F620,52
,5,F720,60,5,F820,68,5,F920,76,5,FA20,84,5,FB20,92,5,626B
,100,5
 670 DEFPROCinitialise
 680 RESTORE730:C%=242:REPEAT
 690   VDU23,C%
 700   FORB%=1TO8:READH$
 710     VDU EVAL("&"+H$):NEXT
 720   C%=C%+1:UNTILC%=255
 730 DATA7C,C0,C0,EF,CD,CD,CD,0F
 740 DATA7C,C0,C0,E6,C6,C6,C6,0F
 750 DATA7C,C0,C0,EE,C2,C4,C8,1F
 760 DATA7C,C0,C0,E6,C9,C3,C9,06
 770 DATA7C,C0,C0,EC,C4,DE,C4,04
 780 DATA7C,C0,C0,EF,C8,CF,C1,06
 790 DATA7C,C0,C0,E7,C8,CE,C9,06
 800 DATA7C,C0,C0,EF,C1,C2,C4,04
 810 DATA7C,C0,C0,E6,C9,CF,C9,06
 820 DATA7C,C0,C0,EF,C9,CF,C1,01
 830 DATAC7,A4,94,97,94,94,A4,C7
 840 DATA66,89,89,89,89,89,89,66
 850 DATA3C,42,9D,A1,A1,9D,42,3C
 860 DIMA%20:X%=A% MOD256:Y%=A% DIV256
 870 $A%="K.":A%=A%+2
 880 FORK%=0TO9
 890   $A%=STR$(K%)+"|!|"+CHR$(&F2+K%)
 900   CALL&FFF7
 910   NEXT
 920 G%=10:*FX4,1
 930 *FX9,5
 940 *FX10,5
 950 DIMP$(1),M%(G%),CHANGE_VECTOR 100
 960 P$(0)="1. Random key practice"
 970 P$(1)="2. Typing in BASIC text"
 980 S%=70:C=S%/8:I%=0
 990 *TV255,1
1000 ENDPROC
1010 DEFPROCMENU
1020 COLOUR2:PRINT" There are two parts to this program:
"
1030 PRINTTAB(9,2)P$(0)
1040 PRINTTAB(9,4)P$(1)
1050 PRINTTAB(3,8)"Which part do you want, 1 or 2 ?":COL
OUR3
1060 PRINT''CHR$254;" 1983 James Tyler";TAB(35,8);
1070 *FX15,0
1080 REPEAT:A=GET-48
1090   UNTILA=1OR A=2
1100 PRINTTAB(27,8);A;SPC7;TAB(9,A*2)P$(A-1);
1110 I=INKEY(1000):ENDPROC
1120 DEFPROCspc_bar
1130 COLOUR3:PRINT'SPC7"Press <SPACE BAR> to begin";
1140 REPEAT:I=GET:UNTILI=32:CLS
1150 ENDPROC
1160 DEFPROCPART1
1170 PRINT'"  This 1st part will help you to learn"
1180 PRINT"  relative positions of the keys using"
1190 PRINT"  the diagram above."
1200 PRINTSPC5"When a key on the screen flashes,"
1210 PRINT"  try to press the same one on the key-"
1220 PRINT"  board, without looking down !"
1230 PRINTSPC5"The flashing will only stop if you"
1240 PRINT"  hit the correct key - so if you miss,"
1250 PRINT"  keep on trying until you succeed..."
1260 PROCspc_bar
1270 VDU19,1,15;0;
1280 CLS:RF%=0
1290 MC%=0
1300 K%=0
1310 MF%=0
1320 IF RF%=0 R%=RND(48)+47 ELSER%=M%(RND(MC%)-1)
1330 PROCKEY(R%,1)
1340 X1%=X%:Y1$=Y$
1350 *FX15,0
1360 PRINTTAB(20,5)"?";:VDU8
1370 KEY%=GET
1380 IFMF%=1 Y$=Y2$:X%=X2%:GCOL0,3:PROCkey
1390 IF (KEY%<>u%)AND(KEY%<>l%)THEN1480
1400 SOUND1,-10,200,5
1410 Y$=Y1$:X%=X1%:GCOL0,3
1420 PROCkey
1430 IF K%<G% K%=K%+1:GOTO1310
1440 IF RF%=1 G%=10:GOTO1280
1450 PW=(MC%/K%)*100
1460 IF PW<20 THEN1290
1470 RF%=1:G%=MC%*2:PROCFLASH:GOTO1300
1480 SOUND1,-10,50,5
1490 PROCKEY(KEY%,0)
1500 X2%=X%:Y2$=Y$
1510 IF MF%=1 THEN 1350
1520 IF RF%=0 M%(MC%)=R%:MC%=MC%+1
1530 MF%=1:GOTO1350
1540 DEFPROCFLASH
1550 PRINTTAB(5,2)"Try these again..."
1560 *FX9,20
1570 *FX10,20
1580 VDU19,3,15;0;
1590 TIME=0:REPEAT:UNTILTIME>120
1600 VDU19,3,7;0;
1610 *FX9,5
1620 *FX10,5
1630 ENDPROC
1640 DEFPROCPART2
1650 PRINT'"  This part of the program will enable"
1660 PRINT"  you to improve your touch typing by"
1670 PRINT"  copying lines  of BASIC text printed"
1680 PRINT"  by the  computer.Only correct entries"
1690 PRINT"  will be accepted, so there is no need"
1700 PRINT"  to use the <DELETE> key."
1710 PRINTSPC5"When you have finished typing in"
1720 PRINT"  a line, press <RETURN> as usual."''
1730 PROCspc_bar
1740 PROCASSEMBLE:H%=TOP-PAGE-50
1750 CLS:VDU19,1,15;0;17,3
1760 IF C%>39 THEN1830
1770 COLOUR2
1780 PRINTTAB(0,7)"Typing accuracy:"
1790 PRINT''"Typing speed:"SPC5"characters per minute"
1800 COLOUR3
1810 PRINTTAB(17,7);100-INT((MC%/len%)*100);"%"
1820 PRINTTAB(14,9);INT(len%*(60/T%))
1830 PRINTTAB(0,2);
1840 CALL CHANGE_VECTOR
1850 PROCprintline
1860 CALL RESET_VECTOR
1870 MF%=0:MC%=0:C%=TRUE
1880 VDU31,0,4,63,8:TIME=0
1890 *FX15,0
1900 KEY%=GET
1910 IF MF%=0 THEN1970
1920 PROCsave_csr
1930 PRINTTAB(22,6)SPC18;TAB(pos%,vpos%);
1940 VDU19,1,15;0;18,0,3
1950 X%=X1%:Y$=Y1$:PROCkey
1960 X%=X2%:Y$=Y2$:PROCkey:MF%=0
1970 P%=POS
1980 IFP%<>len%THEN2000ELSET%=TIME/100
1990 IFKEY%=13THEN1750ELSENC%=13:GOTO2030
2000 NC%=?(line+P%)
2010 IFKEY%=NC% PRINTCHR$KEY%;:GOTO1890
2020 SOUND1,-10,30,10
2030 PROCKEY(KEY%,0)
2040 X1%=X%:Y1$=Y$
2050 PROCKEY(NC%,1)
2060 X2%=X%:Y2$=Y$
2070 IF(X1%<>X2%)OR(Y1$<>Y2$)THEN2110
2080 PROCsave_csr
2090 VDU19,1,10;0;17,1:PRINTTAB(22,6)"CAPS LOCK CORRECT?
'TAB(pos%,vpos%);
2100 COLOUR3
2110 MF%=1
2120 IFC%<>P% MC%=MC%+1:C%=P%
2130 GOTO1890
2140 DEFPROCprintline
2150 B%=PAGE+RND(H%)
2160 REPEAT:B%=B%+1
2170   UNTIL?B%=13:C%=?(B%+3)-4
2180 IFC%>30THEN2150 ELSEB%=B%+3
2190 PRINT;RND(1000)*10;" ";
2200 FORD%=1TOC%
2210   A%=B%?D%
2220   IFA%>31 ANDA%<>127 CALL&B53A ELSEPRINT;" ";
2230   IFA%=&E5 ORA%=&8CPRINT;RND(1000)*10;:D%=D%+5
2240   NEXT:len%=POS
2250 ENDPROC
2260 DEFPROCASSEMBLE
2270 FORK%=0 TO 2STEP2
2280   P%=CHANGE_VECTOR
2290   [OPTK%
2300   LDA&20E:STA OSWRCH
2310   LDA&20F:STA OSWRCH+1
2320   LDA#STORELINE MOD256:STA&20E
2330   LDA#STORELINE DIV256:STA&20F
2340   LDA#line MOD256:STA&70
2350   LDA#line DIV256:STA&71
2360   LDA#0:STA&72
2370   RTS
2380   .STORELINE STY&73:LDY&72
2390   STA(&70),Y
2400   INY:STY&72:LDY&73
2410   JMP(OSWRCH)
2420   .OSWRCH NOP:NOP
2430   .RESET_VECTOR
2440   LDA OSWRCH:STA&20E
2450   LDA OSWRCH+1:STA&20F
2460   RTS
2470   .line
2480   ]:NEXT:ENDPROC
2490 DEFPROCsave_csr
2500 pos%=POS:vpos%=VPOS:ENDPROC
2510 IFERR=17 THEN160
2520 IF NOT(ERL=600 ANDERR=26)THEN2550
2530 ON A GOTO1350,1890
2540 MODE7
2550 REPORT
2560 PRINT" at line ";ERL
2570 *FX4,0
```

Don Thomasson

PROBLEM PAGE

Part 5 of our series for the confirmed puzzler shows how to sort items into sets, and poses a magic square problem for you to solve.

Last month, you were invited to work out a program to solve logic problems involving the arrangement of items into sets. There is a method for doing this by hand which is the basis of the program. It involves a special kind of table.

The form of table for three sets of three items is shown in Fig. 1. A '1' indicates a positive link, a '0' indicates that a link is barred. Two positive links have been inserted, for A1/B2 and A2/C3. These automatically bar A1/B1, A1/B2, A2/B3, and so on. Only one '1' can occur in a line within a box. The other entries on that line must be '0'.

In addition, B2/C3 is barred, because the entries in the A2 line must match those in the C3 line. If there were two '0' entries in a line within a box, the third entry must be '1'.

The program in listing 1 expands this scheme to the four sets of six items quoted in the set problem. The fact that it is written in modular form was a matter of convenience, not a concession to academic views!

Subroutine 2500 clears the main working array from which the display is refreshed. Subroutine 3000 then allows the item names to be set up, displayed, and corrected if necessary. In the given puzzle, the house numbers should be put into the last set, as there is no room to show them in the display.

Subroutine 1000 then invites input of linkage data. Three operators are used:

- A1=B1 will link items A1 and B1 as being in the same set.
- A1<>B1 will state that the items are in different sets.
- A1#B1 will clear an entry (but not its consequences).

Subroutine 1200 analyses the input line into two names and an operator. The operator determines the value of OP, and the names are converted into pointers to the main array, R1 and R2.

Subroutine 1400 then comes into play. R1, which sets the 'line' of the array, must be less than R2, which sets the column. The two pointers are exchanged, if necessary, to achieve this. Then, for OP=0 a full stop is set in the display and 1 is set in the main array, using subroutine 1600. For OP=2, '0' is set in the display and 5 in the array. All new entries are 'bright' to make them stand out.

For the above actions, no consequent action is taken. It would be possible to search for '1' entries in the R1 and R2 lines, and enter the 'reflections' of the single entries, but that is not strictly essential, especially if barred entries are made first. If an error is made, however, it may be best to input 'WIPE', which clears the main array, and start again.

For OP=1, the action is more complex. If there is already a '0' entry at the chosen point, 'ERROR' is reported, and the routine drops out. Otherwise, subroutine 1700 is called to set up the six directly-barred entries in line with the '1' sign. The entries in the lines defined by R1 and R2 are then compared. If they are the same, no action is necessary. If either is '.', it is set to match the other. If one is '1' and the other '0', ERROR is reported.

Because any new '=' entry may have far-reaching consequences, its co-ordinates are stacked in array D to make sure no entry is lost, but that becomes more significant in the final stage.

When an entry has been completed, subroutine 2000 refreshes the display, removing the bright entries.

The final function is called by an input of SCAN, which invokes subroutine 2200. This may be called at any time, but its performance is shown to best advantage if all the given data is input first. You can then sit back and watch the fun.

SCAN locates any lines of the .000 form, where the blank entry must be '1'. Once this process starts, it should go on step by step until the diagram is complete, with all the relationships defined. If it stops short, some essential item of information is missing, and this is very useful in compiling puzzles of this type.

The program has been described as briefly as possible in order to leave room for the listing and the problem for next month. You are invited to teach your computer to work out magic squares for which partial data is given, as follows:

x	x	x	12
8	x	2	x
x	x	15	x
x	3	x	5

Each row, column and main diagonal must add up to 34, and the final diagram should contain the numbers 1 to 16.

	B1	B2	B3	C1	C2	C3
A1	0	1	0			0
A2		0		0	0	1
A3		0				0
B1						
B2						0
B3						

Fig. 1 Truth matrix for three sets of three items.

```
100 DIM A$(6,4,8): DIM P$(20): DIM Q$(20): DIM C(24,24): DIM D(60,2)
110 GO SUB 2500: REM ** CLEAR ARRAY
120 GO SUB 3000: REM ** INPUT NAMES
130 CLS
140 GO SUB 2000: REM ** PUT UP DISPLAY
150 GO SUB 1000: REM ** INPUT LINKAGE DATA
160 GO SUB 1400: REM ** MODIFY DATA
170 GO TO 140
1000 INPUT "DATA ? ";P$
1010 PRINT AT 20,0,"[10 SPC]"
1020 IF P$(1 TO 4)="WIPE" THEN GO SUB 2500: GO SUB 2000: GO TO 1000
1030 IF P$(1 TO 4)="SCAN" THEN GO SUB 2200: GO SUB 2000: GO TO 1000
1040 LET K=1: LET L=1: LET Q$="": LET A=0: LET B=0
1050 GO SUB 1200
1060 IF A*B=0 THEN PRINT AT 20,0,"NOT FOUND": GO TO 1000
1070 LET R1=4*(A-1)+B: LET L=1: LET A=0: LET B=0: LET Q$=""
1080 GO SUB 1200
1090 IF A*B=0 THEN PRINT AT 20,0;"NOT FOUND": GO TO 1000
1100 LET R2=4*(A-1)+B
1110 RETURN
1200 IF P$(K)="[SPC]" THEN GO TO 1260
1210 IF P$(K)="#" THEN LET OP=0: GO TO 1290
1220 IF P$(K)="=" THEN LET OP=1: GO TO 1290
1230 IF P$(K)="<>" THEN LET OP=2: GO TO 1290
1240 LET Q$(L)=P$(K)
1250 LET L=L+1
1260 LET K=K+1
1270 IF K>20 THEN GO TO 1300
1280 GO TO 1200
1290 LET K=K+1
1300 FOR T=1 TO 6: FOR N=1 TO 4
1310 IF Q$(1 TO 8)=A$(T,N,1 TO 8) THEN LET A=T: LET B=N
1320 NEXT N: NEXT T
1330 RETURN
1400 LET ERROR=0: LET F=0
1410 IF R1>R2 THEN LET R3=R1: LET R1=R2: LET R2=R3
1420 IF OP=0 THEN LET M=1: LET A=R1: LET B=R2: GO TO 1600
1430 IF OP=2 THEN LET M=5: LET A=R1: LET B=R2: GO TO 1600
1440 IF C(R1,R2)=5 THEN LET ERROR=1: GO TO 1460
1450 GO SUB 1700
1460 IF ERROR=1 THEN PRINT AT 20,0,"ERROR": RETURN
1470 LET A=R1: LET B=R2: LET M=20: GO SUB 1600
1480 IF F=0 THEN RETURN
1490 LET R1=D(F,1): LET R2=D(F,2): LET F=F-1
1500 IF R1>R2 THEN LET R3=R1: LET R1=R2: LET R2=R3
1510 GO TO 1440
1600 LET C(A,B)=M
1610 IF M=1 THEN LET E$="."
1620 IF M=5 THEN LET E$="0"
1630 IF M=20 THEN LET E$="1"
1640 LET P=7: LET U=INT ((A-1)/4): IF U=1 OR U=3 THEN LET P=5
1650 PRINT BRIGHT 1;PAPER P;AT A,B+INT ((B-1)/4+1);E$
1660 RETURN
1700 LET S=4*INT ((R1-1)/4)
1710 FOR X=S+1 TO S+4
1720 IF X=R1 THEN GO TO 1740
1730 LET A=X: LET B=R2: LET M=5: GO SUB 1600
1740 NEXT X
1750 LET S=4*INT ((R2-1)/4)
1760 FOR X=S+1 TO S+4
1770 IF X=R2 THEN GO TO 1790
1780 LET A=R1: LET B=X: LET M=5: GO SUB 1600
1790 NEXT X
1800 LET BR1=4*INT ((R1-1)/4): LET BR2=BR1+5
1810 LET BR3=4*INT ((R2-1)/4): LET BR4=BR3+5
1820 LET H2=R1: LET H4=R2
1830 IF BR1<4 THEN GO TO 1850
1840 FOR X=1 TO BR1: LET H1=X: LET H3=X: GO SUB 1900: NEXT X
1850 LET H1=R1: IF BR3<BR2 THEN GO TO 1870
1860 FOR X=BR2 TO BR3: LET H2=X: LET H3=X: GO SUB 1900: NEXT X
1870 LET H3=R2: IF BR4>21 THEN GO TO 1890
1880 FOR X=BR4 TO 24: LET H2=X: LET H4=X: GO SUB 1900: NEXT X
1890 RETURN
1900 IF C(H1,H2)=C(H3,H4) OR ERROR=1 THEN RETURN
1910 IF C(H1,H2)+C(H3,H4)=25 THEN LET ERROR=1: RETURN
1920 IF C(H1,H2)=1 AND C(H3,H4)=5 THEN LET A=H1: LET B=H2: LET M=5: GO TO 1600
1930 IF C(H1,H2)=5 AND C(H3,H4)=1 THEN LET A=H3: LET B=H4: LET M=5: GO TO 1600
1940 IF C(H1,H2)=1 AND C(H3,H4)=20 THEN LET A=H1: LET B=H2: GO TO 1970
1950 IF C(H1,H2)=20 AND C(H3,H4)=1 THEN LET A=H3: LET B=H4: GO TO 1970
1960 RETURN
1970 LET F=F+1: LET D(F,1)=A: LET D(F,2)=B: RETURN
2000 FOR T=1 TO 5
2010 IF T=2 OR T=4 THEN PAPER 5
2020 IF T=3 OR T=5 THEN PAPER 7
2030 FOR N=1 TO 4
2040 LET TAB=5*(T-1)
2050 LET LINE=4*(T-1)+N
2060 PRINT AT LINE,TAB;A$(T,N,1 TO 7);
2070 FOR X=0 TO 5-T
2080 FOR Y=1 TO 4
2090 LET G=C(LINE,4*(X+T)+Y)
2100 IF G=1 THEN PRINT ".";
2110 IF G=5 THEN PRINT "0";
2120 IF G=20 THEN PRINT "1";
2130 NEXT Y
2140 PRINT "[SPC]";
2150 NEXT X
2160 NEXT N
2170 NEXT T
2180 RETURN
2200 LET FH=0: FOR A=0 TO 5: FOR B=A TO 5
2210 LET S1=4*A: LET S2=4*B
2220 FOR C=1 TO 4
2230 LET J=0: LET K=0
2240 FOR D=1 TO 4
2250 LET J=J+C(S1+C,S2+D)
2260 LET K=K+C(S1+D,S2+C)
2270 IF J<>16 THEN GO TO 2320
2280 FOR E=1 TO 4
2290 IF C(S1+C,S2+E)=1 THEN LET F=F+1: LET D(F,1)=S1+C: LET D(F,2)=S2+E
2300 LET FH=1: GO SUB 1470
2310 NEXT E
2320 IF K<>16 THEN GO TO 2370
2330 FOR E=1 TO 4
2340 IF C(S1+E,S2+C)=1 THEN LET F=F+1: LET D(F,1)=S1+E: LET D(F,2)=S2+C
2350 LET FH=1: GO SUB 1490
2360 NEXT E
2370 NEXT D
2380 NEXT C
2390 NEXT B
2400 NEXT A
2410 IF FH=1 AND ERROR=0 THEN GO TO 2200
2420 RETURN
2500 FOR X=1 TO 24: FOR Y=1 TO 24
2510 LET C(X,Y)=1
2520 NEXT Y: NEXT X
2530 RETURN
3000 CLS
3010 FOR T=1 TO 6: FOR N=1 TO 4
3020 PRINT AT 0,0;"SET ";T;" NAME ";N
3030 INPUT "NAME ? ";A$(T,N)
3040 NEXT N: NEXT T
3050 PRINT AT 0,0;"INPUT COMPLETE"
3060 FOR T=1 TO 6: FOR N=1 TO 4
3070 PRINT AT 2*T,8*(N-1);A$(T,N)
3080 NEXT N: NEXT T
3090 INPUT "ANY CORRECTIONS ? ";N$
3100 IF N$="N" OR N$="n" THEN RETURN
3110 INPUT "TYPE ? ";T
3120 INPUT "NUMBER ? ";N
3130 INPUT "NAME ? ";A$(T,N)
3140 GO TO 3060
```

Henry Budgett

CT STANDARDS

Our regular page explaining the meaning of the various symbols we use to make programs portable.

It has been very encouraging to see the number of programs submitted using our standard codes for graphics and other non-printable characters. However, it has also become increasingly clear that some of our readers haven't heard of them and this page is intended to set them out once again.

All standards tend to be irksome to adhere to but the ones laid out here are fairly simple and tend to make software easier to maintain by the programmer and simpler to understand for others.

CONTROL THAT CURSOR

Our original standards have now grown with the times. Machines such as the Commodore VIC which have a dual Shift capability can now be incorporated, as can those systems which use Control key functions.

The recently introduced BBC system offers pre-programmed function keys which, we are glad to say, can also be handled by our original coding system. It's nice to see just how well adapted the original standards have become over the last two years! (Indeed, a whole series of books is using them as its *de-facto* standard.) The standards for the cursor controls are given in Fig. 1.

```
[CLS]    CLear Screen
[HOM]    HOMe cursor
[CL]     Cursor Left
[CR]     Cursor Right
[CU]     Cursor Up
[CD]     Cursor Down
[REV]    REVerse video on
[OFF]    Turn it OFF
[SPC]    SPaCe
[CTL]    ConTroL key
[fn]     Function key (BBC)
[G<]     Graphic left (VIC/MZ-80A)
[G>]     Graphic right (VIC/MZ-80A)
```

Fig. 1. Our extended set of cursor control standards includes four new functions.

To indicate more than one of the above, an optional number can be placed within the brackets; [4 CL], etc.

The use of square brackets has raised one or two queries. The reason for this choice is that *most* of the common microcomputer BASICs don't use them for specific functions. In fact, at least one machine provides an added bonus by returning a Syntax Error if they are found, a useful check in case you type them in by mistake.

The code [SPC] was added to the list of cursor control codes to get over the problem of indicating just how many spaces are contained in the gap in the printout. The other common variant of the code for spaces is used by the ZX people. Their choice was '⍽' and this crops up in the various newsletters they publish.

The code [RVS] has caused a few headaches. This is really specific to the PET where the character set can be displayed in reversed video. On machines which don't have this facility you should either find a character in the set which is the reversed image of the one you want and use that or simply ignore it and use anything else you fancy! Don't forget, you may have to look up and alter the values used elsewhere in the program.

THE GRAPHIC SOLUTION

It soon became obvious that the techniques applied to the confusing cursor controls could also be applied to the graphics symbols. The following standard is now in general use in programs published in *Computing Today*.

If a graphics character or characters are to be displayed in a listing (as opposed to POKE codes or CHR$( ) codes) then they are indicated by the method shown in Fig. 2.

Several people have asked what the relationship between the POKE value for a character and that of its shifted graphic might be. In general the shifted version of any character will be 64 greater than the value of that character. This applies to both PET and MZ-80K systems in all cases.

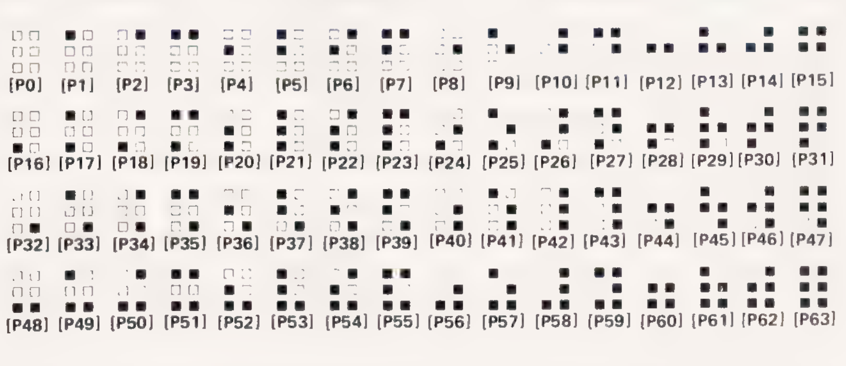


Fig. 3. The standard pixel codes; they will work on most computers which employ this technique as well as for Teletext and Prestel.

This can be taken further to include machines which use a pixel graphics set rather than pre-programmed PET-style characters and the series of codes for these is given in Fig. 3. As is nearly always the case there is one machine to which the standard shown in Fig. 3 does not apply — Tangerine's Microtan/Micron. This machine uses a four by two cell structure for its pixel graphics instead of the Prestel/Teletext three by two cell. The method for calculating the value to assign to 'P' is shown in Fig. 4, and is fortunately nice and simple.

MAKING REMARKS

Many people scorn the use of REMs within programs but, during the development at least, they are extremely useful. One of the documentation methods that we use is to keep our back-up copy of our programs on a 300 Baud CUTS tape with all the REMs in place: the working copy, be it on tape or disc, is REMless in order to save space.

It is also good programming 'manners' to give your REMs odd line numbers:

```
3999 REM ** CRASH PROOF INPUT
4000 INPUT "THE NUMBER OF ENTRIES ";A
```

A remarkable number of submitted programs have jumps that go not to the relevant point in the program, but to the REM statement. This can cause severe problems when re-numbering after removing the REMs.

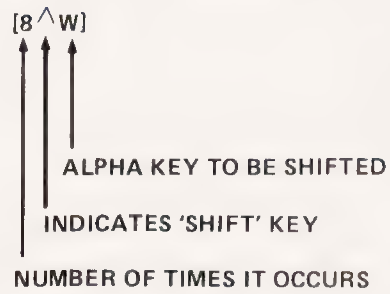


Fig. 2. The way we indicate block graphics on machines like the PET and Sharp. The VIC system of Shift Left and Shift Right is shown in Fig. 1.

1	2
4	8
16	32
64	128

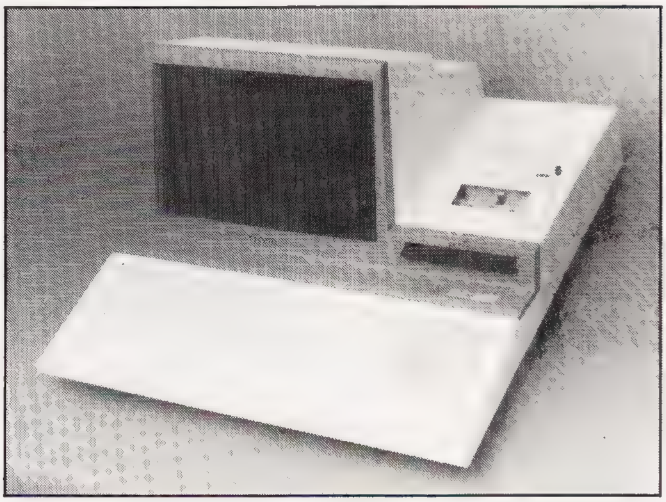
Fig. 4. To convert a Tangerine pixel code into its blocks, simply decode the number into its binary or Hex value and fill in the relevant squares.

SHARP MICRODEALER

SHARP MZ-80A £549

MEMORY	48K RAM	4K ROM		
LANGUAGE	Microsoft BASIC			
CASSETTE	1200 baud (built-in)			
DISC	extra	DOS		
KEYBOARD	QWERTY ☑	CURSOR ☑	NUMERIC ☑	FUNCT ☐
DISPLAY	TV ☐	MONITOR ☑	SUPPLIED ☑	
INTERFACE	PARA ☑	SERIAL ☐	BUS ☑	
GRAPHICS	BLOCK ☑	USER ☐		
	LINE ☐	RES 80 by 50		
	COLOUR	TEXT 25 by 40		
SOUND	Single channel			

Notes: The Sharp MZ-80A is a Z80 based micro. An expansion unit, printer, floppy disc unit and other peripherals are available. Other languages can also be used such as Pascal merely by replacing the tape. With the floppy disc option the machine can respond to higher level software such as Disc BASIC and FDOS (including BASIC compiler). A small range of business and educational software is available. The supplier is **Sharp Electronics (UK) Ltd.** Thorp Road, Newton Heath, Manchester M10 9BE.



SHARP MZ-80B £899

MEMORY	64K RAM	2K ROM'		
LANGUAGE	BASIC (on tape)			
CASSETTE	1800 baud built-in			
DISC	extra	DOS		
KEYBOARD	QWERTY ☑	CURSOR ☑	NUMERIC ☑	FUNCT ☐
DISPLAY	TV ☐	MONITOR ☑	SUPPLIED ☑	
INTERFACE	PARA ☐	SERIAL ☐	BUS ☑	
GRAPHICS	BLOCK ☑	USER ☐		
	LINE ☑	RES 320 by 200		
	COLOUR	TEXT 25 by 80		
SOUND	3 channels			

Notes: The Sharp MZ-80B is a Z80A based micro. Various other languages can be loaded as the machine is "soft", no language being fitted in ROM. Expansion unit, the MZ-80P5 printer and the MZ-80FB floppy disc drive are also available. The supplier is **Sharp Electronics (UK) Ltd.** Thorp Road, Newton Heath, Manchester.

Garry Marshall

BOOK PAGE

This month our reviewer takes a look at some books that run the whole gamut of computing, not from A to B but from CBM to IBM.

In their own ways, this month's books cover the complete range of personal computing. There is a book for the Commodore 64, which is a good representative of the micros with colour graphics and sound capabilities at the cheaper end of the market. A book of CP/M represents all the machines using this operating system from the Osborne 1, at £1581, to as far up-market as you might care to go. Also, of course, machines such as the Apple and the BBC micro can run CP/M when a Z80 or 8085 card is added to them. However, the combined price of the basic machine, the second processor, the disc unit and the software is unlikely to be less than the cost of an Osborne 1. Two books on the IBM personal computer represent the top end of the market, where this machine, at £2392, is firmly placed. It seems that in the States this computer can be regarded as a games machine as well as a vehicle for serious software, but until the levels of disposable income rise considerably in this country it is unlikely to enjoy such an image here.

The **Commodore 64 Games Book** is a compilation of listings of games programs. The games are mostly quite familiar, and include such well-known titles as 'Flight Simulator', 'Fruit Machine', 'Luna Landa' (sic) and 'Invaders'. There are 30 listings in all. To dispense with the criticisms of the book first, it is reproduced directly from a mixture of listings from a dot-matrix printer and typing with at least two different type faces, which leaves a rather scrappy impression. The explanations of the programs are minimal which is especially unfortunate with this machine, where certain effects are achieved by POKE-ing and PEEKing particular locations. The reproduction of the illustrations in the text is extremely muddy, as is easily appreciated by a quick comparison with the much smaller colour photos on the glossy back cover. Lastly, the program listings include within character strings the special Commodore graphics characters for representing cursor movements, clearing the screen and other similar things. In my view, ***Computing Today's*** convention for representing these things makes program listings much easier to read and to key in (see page 78.

Having said this however, a book of this kind stands or falls by the quality of its programs, and there is nothing wrong with the programs given here. They provide the owner of a 64 with the opportunity to key in all these games, learn a good deal about the computer in doing so, and then to enjoy playing the games. They make good use of the 64's capabilities, but programming activities such as the sprite graphics, which require the use of POKE with special locations, would be much easier to understand and to adapt if fuller documentation were given.

All the programs in the book are available on cassette for anyone who wants to play the games but can't be bothered to type them in. The publishers do not give the price, but it is presumably unlikely to be much above the price of the book. This raises an interesting point. If 30 games can be made available in a book, or on cassette, for £5.95 (that is at five for a pound) why do most of the off-the-shelf games cost so much more? Admittedly, the latter are generally more sophisticated than the games in this book, but the comparison does make them seem over priced. Further, if off-the-shelf software is overpriced, or even if one only thinks that it is overpriced, one is less likely to condemn its being pirated.

As a final thought in connection with this book, now that the portable version of the 64 has been seen at the Commodore show, wouldn't it be nice to have a cheap book of 'serious' software for it?

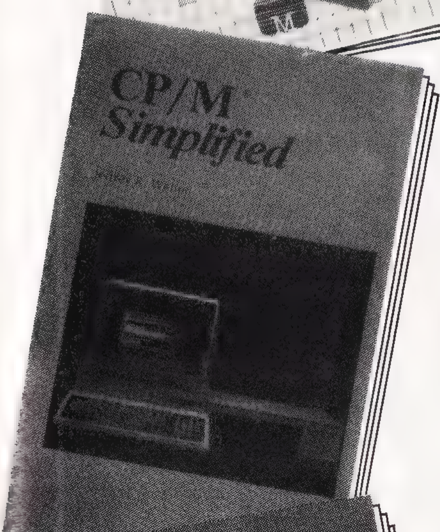
I looked forward to reading **CP/M Simplified** since, in my view, the need for a really simple introduction to CP/M has never been met. Unfortunately, it doesn't fill the gap and is, to me, less satisfactory than Rodney Zaks' book on CP/M which is the best that I have read.

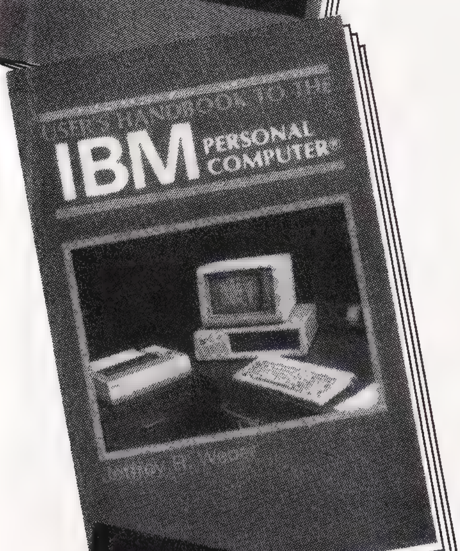
There is a difficulty with books on CP/M in that it is easy to transfer one's dissatisfaction with CP/M generally to the author trying his best to write a book about it. But the failing of introductory books on CP/M is that they plunge into explanations of everything that it can do without managing to explain to the beginner why he should want to do them. This approach is fine for those who already understand and use CP/M, or any other operating system, but if fails to motivate the beginner. Weber's book also adopts exactly this approach.

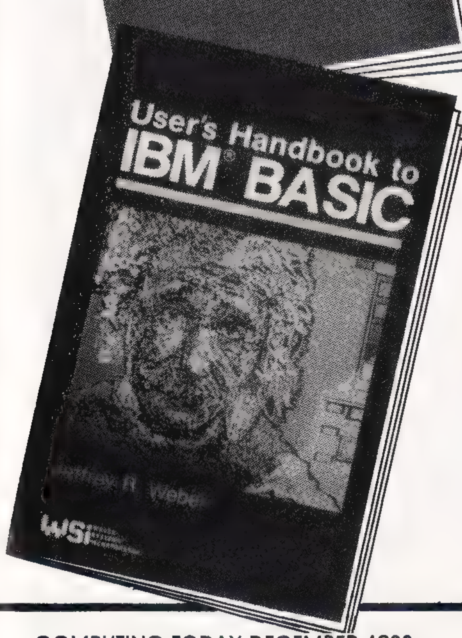
The book begins with a very brief introductory chapter, then introduces business computing under CP/M and a range of business packages, all of which are available from Weber Systems Inc (remember the author's name?) before moving on to the basic uses of CP/M. The main impression of these chapters is that the author really cannot wait to get the introductory material finished with so that he can move on to the more substantial matters.

The chapter on the basic uses of CP/M does introduce most of the commands and special functions by giving examples such as how to copy a file and a whole disc. An example of how to run a program under CBASIC does show why it is necessary to be able to copy files, although it may also cause the newcomer to wonder if there is no simpler way to run a BASIC program. To be fair, the author mentions that there is, and even goes on to explain what compiled BASIC is and what merits it has. This example shows that it is possible to give examples that provide motivation as well as illustrating the difficulty of devising simple examples for this purpose. Brief explanations of the commands DIR, REN, ERA and ED are also given so that most of the transient commands are covered. The importance of caring for discs, making copies of them and of labelling them properly is stressed. Good explanations of the way the system prompts work and of cold and warm starts are given, although the ideas of the transient commands and command files need a fuller treatment to overcome the difficulty that many people seem to have in grasping their purposes.

Chapter 4 deals in more detail with CP/M's special control characters and the commands for handling files, devices and programs. This is really a reference chapter and is, as such, quite well done. Chapter 5 examines MP/M and CP/M version 2.2. MP/M is an operating system that can handle several programs at the same time, while version 2.2 of CP/M has several features that were designed for MP/M. I imagine that the reader who is new to CP/M will gladly skip this chapter, but it does provide a useful source of reference for those who need it. The following chapters







go into considerable detail about file handling with PIP, editing with ED and using CBASIC, while Chapter 9 provides a detailed reference guide to all the features of CP/M and MP/M.

The book is a useful source of reference to CP/M, and on this score I would not fault it. However, I do not think that it simplifies CP/M, even compared to books that make no claim to do so. Its introductory chapters are clearly the weakest, and this is a pity because a satisfactory introduction for beginners would be very valuable.

The **User's Handbook to the Personal Computer,** also by J.R. Weber, is exactly what its title suggests, and it is a rather good handbook at that. In fact, it covers the entire system, and not just the personal computer itself, including the printer. It has chapters on the hardware, on installing the computer, on operating it, as well as on its disc operating system (DOS) and its BASIC. This is precisely what one would hope to find in the handbook supplied with the computer, especially as it is pitched at a level that even the least computerate (if you will excuse the word) can appreciate. But herein lies a puzzle, for surely the IBM manuals supplied with the computer cover the same ground in an entirely suitable way. So the puzzle is, for whom is this book intended? Perhaps it is aimed at anyone considering the purchase of an IBM PC, although I expect that IBM will sell you their manuals and, in any case, wouldn't a (free) demonstration be much more informative?

Anyway, leaving these matters aside, the book does give a good idea of what the IBM PC consists of, what it can do and how it can be used. The hardware description gives all the information that one could want on everything from the Intel 8088 microprocessor at the heart of the system to the details of the construction of the printer intended for use with the system. The chapter on installation gives the most comprehensive 'idiot's guide' that I have ever seen, even including diagrams to show how to remove each unit from the bag it is wrapped in! The treatment of troubleshooting contains such down-to-earth (but none the less valuable for that) information on the little things that ought to be checked if the computer can't be made to run. However, it inevitably turns out that if the computer cannot be made to run because something is really wrong with it, then it must be returned to IBM. This is rather depressing in view of all the built-in diagnostic tests. Wouldn't it be great if it could not only test itself, but also repair itself?

The chapter on using IBM DOS, one of three disc operating systems available for the computer, shows how easy it can be to use a decent DOS. The two descriptions that are given of copying the contents of a disc using a single disc drive and using a double disc drive can be used to provide a convincing illustration for anyone who is not clear whether they need single or dual disc drives.

In the remaining chapters one is continually struck by points of similarity between the IBM PC and other personal computers. These include the pixel graphics characters and codes which are like those on the TRS-80, function keys as on the VIC and Commodore 64 (F7 is TRON, by the way), and the ability to enter BASIC keywords with a single keystroke as per Sinclair.

In summary, it can be said that the book gives a good appreciation of all aspects of the IBM PC. Its final chapters cover the use of its BASIC, and provide a reference guide to it. This is expanded further in the **User's Guide to IBM BASIC** by the same author, which has much material in common with the previous book, but does give a much more detailed treatment of the IBM PC's dialect of BASIC. The version of BASIC is an extended Microsoft BASIC, so that its core will be familiar to many. It provides several graphics commands, including DRAW, PAINT and CIRCLE. It also has commands for bit-manipulation and VARPTR to return the address of the memory area assigned to a variable.

In conclusion, it may be observed that some of this month's books are the kind that you would read from cover to cover. Nevertheless, whether for the programs that they give or as a source of reference, they can add considerably to one's enjoyment and mastery of the computer.

The books reviewed this month are:

Commodore 64 Games Book by Clifford and Mark Ramshaw, Melbourne House Publishers, 187 pages, £5.95
CP/M Simplified by J.R. Weber, Weber Systems Inc, 316 pages, $13.95
User's Handbook to the IBM Personal Computer by J.R. Weber, Weber Systems Inc, 294 pages, $13.95
User's Handbook to IBM BASIC by J.R. Weber, Weber Systems Inc, 309 pages, $13.95

Nearly 400 Programmes IN STOCK

Ring for our Price List

CLWYD PERSONAL COMPUTERS

Unit 19, Daniel Owen Precinct, Mold, CH7 1AP.
Telephone: Mold 56842

ATARI®

400 inc BASIC **£115**
800 **£299**
48K Lynx **£225.00**
COMMODORE 64 £229

VIC STARTER PACK £139.99p

ORIC 1 48K £139.99p

Phone Orders Welcome on **0352-56842**

Postage & Packaging **FREE**

NEXT DAY DELIVERY AVAILABLE Please ring for details:

I enclose cheque/P.O. for £ ____________
or please debit my Access Card No.

NAME ______________________________
ADDRESS ___________________________

University Software

LIBRARY OF ADVANCED MATH/STAT/ECON

TAPE 1: MATRIX OPERATIONS — **SPECTRUM £6.95 ZX81 £5.95**

Side A: Inversion, multiplication, addition, subtraction and scalar multiplication of matrices and vectors within one single program. Any output can in turn be used as the input of the next operation without re-typing. Capacity (no of rows x no of columns), 16K ZX81 25x25, 16K Spectrum: 17x17, 48K Spectrum 48x48
Side B: Determinants of square matrices

TAPE 2: POLYNOMIALS — **SPECTRUM £6.95 ZX81 £5.95**

Side A: Includes quadratic equations (as degree 2 polynomials) and Newton-Raphson and half-interval search methods for higher degree polynomials. Computes the roots with 8 digits of precision
Side B: You can plot polynomials in any interval and examine their roots extremum points

TAPE 3: INTEGRATION — **SPECTRUM £6.95 ZX81 £5.95**

Side A: Integration of functions by Simpson's and trapezoidal rules. Also computes the area enclosed by two functions
Side B: Plot of integrals. Integration can be visualised on the screen

TAPE 4: REGRESSION — **SPECTRUM £7.95 ZX81 £6.95**

Side A: A highly developed multivariate regression program featuring Log/Ln option on each variable (thus allowing exponential and geometric regressions). R^2, corrected R^2 standard errors, t-statistics, F-statistic, degrees of freedom, Durbin-Watson statistic interpolation. Capacity (no of variables x no of observations), 16K ZX81: 2x500, 5x250 10x140, 16K Spectrum. 2x200, 5x100, 10x50, 48K Spectrum 2x1800, 5x900, 10x500
Side B: Plot of bivariate regressions. You can see how your computer draws a best-fitting line on a set of numbered data points

TAPE 5(a): LINEAR PROGRAMMING — **SPECTRUM £7.95 ZX81 £6.95**

Side A: A user friendly optimisation program capable of handling all sorts of linear programming problems (any combination of $\leq, =, \geq$, constraints and $x_i \geq$, $x_i \leq 0$, $-\alpha < x_i < \alpha$ sign constraints). Features the cannonical equivalent of the prima values of slack variables and the dual. Capacity (no of variables x no of constraints), 16K ZX81, 10x23, 15x20, 20x15 16K Spectrum 10x10, 48K Spectrum, 10x50) 25x40 50x30
Side B: Solutions of simultaneous equations

TAPE 5(b): PROFESSIONAL LINEAR PROGRAMMING — **£14.95**

Available for 16K ZX81 and 48K Spectrum with above features plus 'save-data' and 'change-data' facilities. Any single data entry can be changed in order to observe its effect on the solution

• All inclusive prices for the UK •

• Tapes 1-5(a) Spectrum **£35** ZX81 **£30**

Cheques payable to:

UNIVERSITY SOFTWARE
29 St. Peter's Street, London N1 8JP.

The Data Store

6 CHATTERTON ROAD
BROMLEY
KENT

for the BBC MICRO

OFFICIAL ACORN DEALERS

WIDE SELECTION OF SOFTWARE AND PERIPHERAL EQUIPMENT INCLUDING

EPSON, NEC, SEIKOSHA
PRINTERS

ZENITH, CABEL
MONITORS

CUMANA
DISC-DRIVES

BOOKS AND CABLES AVAILABLE
plus our personal advice service

MACHINES DELIVERED & SET UP IN YOUR HOME

PHONE 01 460 8991 (9.30 – 5.30)
ORPINGTON 26698 (Evenings)
(CLOSED WEDNESDAY)

It's easy to complain about advertisements.

The Advertising Standards Authority.
If an advertisement is wrong, we're here to put it right.
A.S.A. Ltd., Brook House, Torrington Place, London WC1E 7HN.

AD INDEX

A B & C 12
AGF 12
AHKTER 27
BICC VERO 4
CGL 5
CALCO 70
CASCADE 63
CLWYD COMPS 82
COMMODORE 42/43
CROWN 13
DATACLONE 33
DATAGRID 83
DATAPEN 75
DATASTORE 82
DRAGON 41
DUCKWORTH 16
ELECTRONEQUIP 66
GILSOFT 12
HAPPY MEMORIES 4
IPC (EVERYDAY ELECS) 4
KUMA 37
LEC 74
LEVEL 9 26
LONGMAN 28
MDA MODON 16
MAP 80 12
MATTEL AQUARIUS 48/49
MAYFAIR MICROS 75
MICROGEN 74
MICROTANIC 69
MICROTANIC (PRINTERACE) 66
MICROTECH (LEEDS) 82
MOLIMERX 21
ORION 13
ORIC SOFTWARE 70
PRE ELECTRONICS 83
PREMIER .... IBC
POWERTRAN CYBERNETICS .... IFC
SP ELECTRONICS 77
SHARP 84/85
SILICA SHOP .... OBC
SINCLAIR 57/60
SINCLAIR 16
SIR COMPUTERS 74
SPECTRA VIDEO 50
SPRAKEFIELD 77
SWANLEY ELECTRONICS 70
SUPERSOFT 75
TECHNEG CLWYD 70
TEXAS 33/35
ULTIMATE 51
UNIVERSITY SOFTWARE 82
VIRGIN 8/9

NO LIMIT

Now, a home computer with virtually no limit to its possibilities. The astonishing new Sharp MZ700. A machine with a dazzling array of talents.

First, it's a 'clean' machine. So you are not limited to any one computer language. You have the flexibility to run and write programs in BASIC, FORTRAN, MACHINE CODE, PASCAL, ASSEMBLER and many others. And the MZ700's 512 predefined characters mean you can build up detailed pictures on the screen, without spending time specifying and designing special characters for games and special effects.

Second, it has a memory of 64K. So as your technique improves and develops, you are able to move forward to more and more advanced programming.

All of which makes this the perfect home computer for parents, as well as children.

The MZ700 gives you access to a wide choice of new software, from only £3·95 per cassette. An additional plotter/printer, costing £129·95, can produce high resolution graphics in 4 colours. A data cassette recorder is an extra at £39·95. Both add-ons fit snugly into this easy to carry compact system with no trailing wires. And you get ten exciting games, free on purchase, including Super Puckman, Circus Star, Snake v. Snake, and Man-Hunt.

The brilliant new MZ700.
The no-limit computer. £249·95. From Sharp. Where great ideas come to life.

CPU:	Z80A		1
ROM:	Monitor	4K byte ROM	1
	Character generator	4K byte ROM	1
RAM:	64K byte D-RAM		8
	4K byte V-RAM		2
1/0 bus:	Expansion 1/0 bus		1
	Additional printer 1/0 bus		1
	Cassette READ/WRITE terminals		2
	Joystick terminals		2
RGB MONITOR/VIDEO MONITOR/RF OUTPUTS			

COMPATIBLE WITH SHARP MZ 700 SERIES

Look for this logo on software denoting Sharp Compatible Software.

To: Sharp Electronics (UK) Ltd,
Home Computer Division,
Sharp House, Thorp Road, Newton Heath,
Manchester M10 9BE. Tel: 061-205 2333

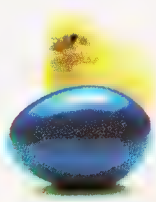
Please send me details of the Sharp MZ700.

Name ________________

Address ________________

CT 2

The world of
SHARP
where great ideas come to life.



LUCAS MICRODEALER

LUCAS LX

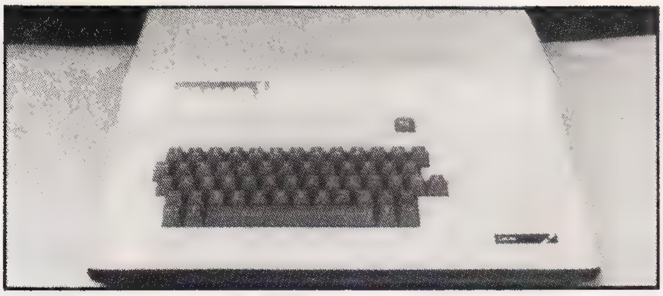
MEMORY	64K RAM expandable to 256K
LANGUAGE	Microsoft BASIC
CASSETTE	300 or 1200 baud
DISC	Single or twin 5¼ floppy disc drives DOS CP/M 2.2 (supplied) or NAS-DOS
KEYBOARD	QWERTY ☑ CURSOR ☑ NUMERIC ☑ FUNCT ☑
DISPLAY	TV ☑ MONITOR ☑ SUPPLIED ☑
INTERFACE	PARA ☑ SERIAL ☑ BUS ☑
GRAPHICS	BLOCK ☑ USER ☑ LINE ☐ RES 392 by 256 COLOUR 8 TEXT 80 by 25

Notes. The Lucas LX is a Z80 microcomputer aimed more at the professional and business user. Hence 5M Winchester disc interfacing is provided. Popular printers may be used with the RS232 serial interface, and a Centronics interface is also provided. There is an additional parallel interface connector for providing up to 16 on/off signals. The monitor supplied as standard is a 12" monochrome version: a colour monitor is also available. The high res colour graphics may be 392 by 256 in eight colours, or 784 by 256 in two colours. A wide range of applications software is available via the CP/M operating system, including Wordstar, Supercalc, and Calcstar.

NASCOM 3 £376

MEMORY	48K RAM 10K ROM
LANGUAGE	Microsoft BASIC
CASSETTE	300 or 1200 baud
DISC	extra DOS CP/M or NAS-DOS
KEYBOARD	QWERTY ☑ CURSOR ☐ NUMERIC ☐ FUNCT ☐
DISPLAY	TV ☑ MONITOR ☑ SUPPLIED ☐
INTERFACE	PARA ☑ SERIAL ☑ BUS ☑
GRAPHICS	BLOCK ☑ USER ☑ LINE ☐ RES 800 by 256 COLOUR 8 TEXT 25 by 80
SOUND	Three channels

Notes. The Nascom 3 is a Z80 based micro. A second version of BASIC and Pascal are also available, as are a cassette recorder and light pen.



COMPUTING TODAY

Lineage: 35p per word.
Semi display: £8.00 per single column centimetre
Ring for information on series bookings/discounts.

All advertisements in this section must be prepaid.
Advertisements are accepted subject to the terms and conditions printed on the advertisement rate card (available on request).

01-437 1002
EXT 305
Send your requirements to:
MARK BECAREVIC
ASP LTD, 145 Charing Cross Road, London WC2H 0EE.

SOFTWARE APPLICATIONS

NASCOM

COLOUR DISTEL Information Display Executive **£9.95 inc.**
Enhanced version controls colour via the PIO and simple interface. DISTEL is an advanced program for control of dynamic teletext type displays with simple keyboard commands for five fonts, pauses, speed changes, blanking, borders, special graphics, overlaying for animation, mimic diagrams and scrolling announcements. Paging system for replaying in any order. Ideal for advertising, lecturing or creative amusement.

ANIMEX Animation Executive **£7.95 inc.**
Fast machine code program for frame animation. Generate partial images with Edifis for rapid composition. Cartoon examples supplied on tape.

EDIFIS Editor for images **£9.95 inc.**
3K machine code image composition program for high res. characters of pixel images for DISTEL and ANIMEX. Comprehensive commands for translations, mirror, logical combinations, fill, multiple scale etc.

ALL PROGRAMS ON HIGH QUALITY TAPE WITH FULL HANDBOOKS

24HR VISA/ACCESS Service **Tel: 031-552-7501/2**

A.D.D. Ltd, 50 West Harbour Rd., Edinburgh EH5 1PU

TRS80/VG 16KLII software. (1) Basicalc 26x26 matrix spreadsheet program. (2) 747 - An instrument flight simulator. (3) Personal Finance - 28 user definable accounts. Full documentation included. £5.95 each or any two for £10. D. Garvin, 23 Fieldway, Liverpool L15 7LU. Wanted - TRS80 software good royalties

BBC copy protected Acornsoft tapes with master key. No more "locked" messages. Unprotected copy can be copied to disc if required. Works on ST*RSH*P C*MM*ND, SN**K*R, etc. Full instructions. Tape £5.00. To: Nicholas Benton, 1 Cow Lane, Steeple. Aston, Oxon.

SPECTRUM TRADE SECRETS Includes making programs unlistable, unstoppable, invisible. How to reveal such programs plus hints/tips. £3. ZX-Guaranteed (office CT), 29 Chadderton Drive Unsworth, Bury, Lancs

BBC MICRO
POOLS PREDICTOR

An easy-to-use, very powerful forecasting program using 6 different methods of prediction based on mathematical analysis of current form **£4.99**

MAYDAY SOFTWARE
181 Portland Crescent, Stanmore, Middlesex HA7 1LR

SOFTWARE LANGUAGES

FORTH
AVAILABLE NOW

Nas-FORTH 1 11 for Nascom 2 etc
Fig-FORTH with FORTH-79 additions Complete with ZX80 assembler and screen editor Ram-Disc simulation, stack display utility, full support of Nas sys routines Extensive manual full system 9K on tape £25 + VAT

PET FORTH 1 11 for 4000 and 8000 series
Fig-FORTH with FORTH-79 additions complete with 6502 assembler and screen editor Ram-Disc simulation for tape version Stack display utility Powerful IEEE control words Extensive manual Tape version £25.00 + VAT Disc version including tutorial disc £75.00 + VAT

xFORTH for CPM systems
FORTH-79 for all Z80 CPM systems Extensive manual All above features and more £75.00 + VAT including tutorial disc

Purchase of our products includes free membership of xFORTH Users Group, and FORTH tutorial disc or tape. Purchase price includes multiple copies. Full range of FORTH books available.

Write or telephone for more details: 0202 764724.

David Husband,
2 Gorleston Road, Branksome, Poole, Dorset BH12 1NW

GO FORTH & * ;

DIY FORTH KITS
Installation manual **£7.00**
How to do it + model + definition + editor
Source code listing for one processor **£7.00**
6502, 6800, 6809, 8080 Z80, 8086 8088 9900, 1802, 68000, Z8000, VAX, Apple] [LSI-11, Eclipse

JUPITER ACE — a very nice FORTH micro now with free Rampack........ **£78.00 + VAT**

FULL FORTHS ON CASSETTE Spectrum 48L w+ editor colour sound **£14.95**
BBC + editor assembler graphics toolkit **£25.00**
Dragon 32 + editor graphics commands **£18.95**

MicroProcessor Engineering Ltd
21 Hanley Road Shirley
Southampton SO1 5AP
Tel 0703 775482

ADD-ONS

UPGRADE YOUR 16K ZX SPECTRUM TO 48K

SIMPLY FITS INTERNALLY WITH NO SOLDERING ON SERIES 2 MODELS. UNLIKE COMPETITORS, WE USE NO REJECT COMPONENTS AND AN ADAPTOR P.C. BOARD TO ACHIEVE RESULTS. DESIGNED BY EX SINCLAIR ENGINEER. MODEL CP48 £23.95 inc. P&P (EXPORT AND UK), EX STOCK.

32K MEMORY EXTENSION ONLY £23·95

Citadel Products Ltd TEL:01-951 1848
DEPTPCT50 HIGH STREET, EDGWARE, MIDDLESEX HA8 7EP

SOFTWARE GAMES

SHARP MZ-80A. Knockout Whist, with instructions for beginners, and Launcher. £4.99. Scotsoft, 16 Grange Road, Edinburgh EH9 1UJ.

SOFTWARE

TRS-80 & VIDEO GENIE
Chaos (16K) **£9.75**
Dodgems (16K) **£9.75**
Frogger (16K) **£9.75**
Kong (16K) **£9.75**
Muncher (16K) **£9.75**
Pool (16K) .... **£7.95)**

SINCLAIR
ZX 81: Grizzly **£3.95**
Pools **£4.00**
Spectrum: Bermuda Triangle 3 (48K Adventure) **£5.50**
Pools **£4.95**

COLOUR GENIE
Chaos **£7.95**
Frogger **£7.95**
Pools **£7.95**

(All prices include V.A.T. & P&P)

Always interested in programs for any of the above machines (25% Royalties given)

KRICON LTD
11 Medway, Hailsham, East Sussex, BN27 3HE (0323) 846110

NASCOM ACTION? Get it with Revenge of Drosophila! Fast Synchronised Animation, Scrolling Maze, 5 different Maze s to survive and 8 Game Variations. Requires 16K 1, 2 or 3 with NASGRA ROM (Ver. 3) Monitor Independent - Free Loader. Specify CUTS or N format. £8 post free from: Garry Rowland, 24 Parsloes Avenue Dagenham RM9 5NX

SHARP MZ-80K 48K over £140.00 software tapes, four books, users notes etc. Buyer must be willing to collect. £320.00 ono (Telephone: 01-472 1331)

UK 101 SOFTWARE — Frogger, Dominoes, Boston, Yahtzi, Connect 4, Lincs, Simon, Fireman, PF21, Pairs, Defuse. Other programs from 4K to 22K, colour or black/white with sounds. Send for details to Dr Gilchrist, 70 Skirbeck Road, Boston, Lincs PE21 6DG

Microtan 65 Eprom Switching Board

Perm any 4 from 16 Software controlled Bareboard + instructions **£19.95** post-free Also available 64 x 25 colour VDU card, 3 pass assembler, word processor etc Large SAE for details

M.P.D., 7 Cedar Close, Grafham, Huntingdon PE18 0DZ.

CHILTERN ELECTRONICS
THE MONITOR SPECIALISTS
Amazing Value in Video Monitors
Professional quality equipment by AGC Corp ideal for your micro Look at these features ★ Superb resolution for graphics, 80 col etc ★ Bandwidth 10hz-22Mhz ★ Green Screen P31 phorsphor ★ Antiglare tube ★ Attractively styled case ★ Composite video mains operation Available in 9" Model HM911 **£85.** 12" Model HM123 **£89.**

FANTASTIC KEYBOARD BARGAIN!! Scoop purchase allows us to offer a 49 key QWERTY layout, ASCII encoded keyboard. Fully cased, used but guaranteed working Full logic diagrams and information supplied. **ONLY £15.** Add carriage £1 85 and VAT Educational discounts Access welcome Same day despatch Call Gary Kent on 02407 71234 **High Street, Chalfont St Giles, Bucks HP8 4HQ. Telex: 262284.**

LIBRARIES

THE MZ80K/A SOFTWARE LIBRARY

Choose from 150 titles. These include exciting Arcade/Adventure games, and Educational/Business applications. Once only membership fee **£5.00.** Hire charges only **90p** per 14 day period Plus your first 5 games FREE. Send now for free illustrated catalogue to

The Yorkshire Software Library, 13 Park Top, Pudsey, Yorkshire LS28 6BY.

CARTRIDGE CITY will rent Atari 400 and 800 owners cartridges at low rates. Discounts on software purchases. Details from Cartridge City, 25 Gaitside Drive, Aberdeen AB1 7BH

HARDWARE

NEWBRAIN

The professional micro computer for the price of the hobby machine. Delivery free. Call us for very special offer. Mail Order Access accepted

Ring us on Stevenage (0438) 812439

anytime for the latest details of NEWBRAIN hardware, software user groups etc.

also available "Newbrain Dissected" — book and tape

New EPSON, CP80 and JUKI PRINTERS

ANGELA ENTERPRISES,
4 Ninnings Lane, Rabley Heath, Welwyn, Herts AL6 9TD.

FUTURE TECHNOLOGIES

(CT) 1 Hollybrook Rd, Shirley, Southampton, SO1 6RB
(Reg. Office. Callers By Appointment)

CBM 64K Computer £198
With Cassette & 2 Blank Tapes £242
VIC 20 Starter Kit £139
Cassette Alone — £44 90 — Tapes £1 each
Spectrum 16K — £99 95. 48K — £129 95
Also Atari, Texas, Accessories, Software
Cheque/P.O. with Order
Send large SAE for Lists (23p Stamp)
Add £1 p&p for Orders under £50
All prices include VAT

6802 MICRO COMPUTER

This micro is designed for control applications both in industry and the home. Two models are available, PIA or VIA. Both are crystal controlled and have a 5V reg. 128 bytes of RAM, 18in/out lines + 2 input lines. The via also has two 16 bit timers and a shift register. The programs are held in EPROM (not included) Types 2716, 2516 and 2732 are accepted, 6862 pia kit £19 95 6802 via kit £21 95 Assembled add £3.00 Please add 15% VAT Discount available on 5+ qty Orders to — Xanar Dept CT, 20 Baldwin Road, King's Lynn Norfolk PE30 4AN

ACCESSORIES

ELKAN ELECTRONICS

NEW! NEW!

NANOS "quick reference" cards — easier to use than the manuals!

Dragon 32 .... **£3.95**
Commodore 64/
Vic-20 .... **£3.95**
Sinclair ZX81 .... **£3.50**

QUICK-SHOT self-centring joysticks — improve your scores!

Dragon 32 .... **£14.95**
(specially developed by Elkan)
Atari 400/800 .... **£12.95**
Commodore 64/
Vic 20 .... **£12.95**

ADD £1 post & packing

ELKAN ELECTRONICS,
Freepost, 11 Bury New Road, Prestwich, Manchester M25 6LZ.
Tel: 061-798 7613.

FREE CATALOGUE

Containing something for most micro users

Software, hardware, dust covers, books, T-shirts, cassette leads, LOADing aids, test cassettes and head demagnetisers etc For Ace, Atom, Atari, BBC, Commodore, Dragon, Electron, Genie, Lynx, Nascom NewBrain, Oric, Sharp, Spectrum, Texas VIC & ZX81

SEND SAE TO **FULCRUM PRODUCTS**
14 STEEP LANE, FINDON, WEST SUSSEX BN14 0UF

PRINTERS & DISKS

EPSON
FX80 360
RX80 247
FX100FT 514
RX80FT 269
MX100FT 3 399
LUCAS
LX80 250
OKI
M80A 190

SEIKOSHA
GP100A 185
GP100VC 185
GP250X 235
GP700A 380
BASF DISKS
5¼" SS SD ... 1 80 ea
5¼" SS/DD .. 2 00 ea
5¼" DS DD .. 2 30 ea

C.W.O ONLY PLEASE ADD VAT. PRICES INCLUDE P&P&I

NMI COMPUTERS
11 WENTWORTH CT, PONTELAND, NEWCASTLE UPON TYNE

SERVICES

T-Shirts for Summer!

"I'm user friendly" "Have you seen my peripherals?" and many more **£3.25** white or blue, S, M, L
ONE PER CENT SCREENS
12 Star Lane, Great Wakering, Essex

2708 EPROM programming service. Copying, erasing. SAE for details J. Bonnett, 91 Woodheyes Road, London NW10.

WANTED

PROGRAMMER

Good programmer (Spectrum, Commodore or BBC) required for partnership in semi-established software business in London.
TEL: 01 - 994 8912
POTENTIAL LOOKS GOOD

FOR SALE

VIDEO GENIE + 32K Expander EG3014 Masses Software. Books. Lower case mod. soundboard excellent £325 0904 470464

NASCOM 2, 32K Zeap, Nasdis Toolkit, Naspen, Kenilworth Case Twin PUS, NAS-SYS3, Software Books. £275. 0904 470464

DISC DRIVE 5¼ inch Snugart SA 400 Full documentation £85. Tel 01-885 1303 Evenings

NASCOM 2 cased 48K Pascal Zeap Nasdis Debug Program. Power games and file handler £330. Phone M'Bro 0642 59701 after 6pm

MICROTAN IS BACK!! A superb 6502 based system. Tanex Keypad. PSU Offers! 01-788 4864

NASCOM 3 48K. Zeap, Nasdis-Debug in ROM Tape Extended Basic Pascal Other Software. Manuals. Realistic Offers Please. 01-340 7479

SHARP M280K 48K complete with basic and games software in original box. Evenings 0524 69869. £225 ONO.

COMPUTING TODAY
CLASSIFIED ADVERTISEMENT — ORDER FORM

If you have something to sell now's your chance! Don't turn the page — turn to us!

Rates of charge: 35p per word per issue (minimum of 15 words). Please state classification and post to **COMPUTING TODAY, CLASSIFIED DEPT., 145 CHARING CROSS ROAD, 145 CHARING CROSS ROAD, LONDON WC2.**

					£5.25
					£7.00
					£8.75
					£10.50
					£12.25
					£14.00
					£15.75
					£17.50

Please place my advert in **COMPUTING TODAY** for . . issues commencing as soon as possible

I am enclosing my Cheque/Postal Order/International Money Order for: (delete as necessary) £..... (Made payable to A.S.P. Ltd)

OR Debit my Access/Barclaycard (Delete as necessary)

All classified advertisements must be paid for in advance.

Please use BLOCK CAPITALS and include post codes

Name (Mr/Mrs/Miss/Ms) ....
(delete accordingly)
Address ....
....
....
Signature .... **Date** ....
Daytime Tel. No. ....

COMPUTAMART

AT A GLANCE...AT A GLANCE...AT A GLANCE...AT A GLANCE...AT A GLANCE...AT A GLANCE...

COMPUTAMART

AT A GLANCE . . . AT A GLANCE . . . AT A GLANCE . . . AT A GLANCE . . . AT A GLANCE . . . AT A GLANCE . . .

COMPUTING TODAY PRESENTS YOUR OWN 'WHERE TO BUY IT' GUIDE.

SUSSEX

TYNE AND WEAR

WALES

WARWICKSHIRE

LOOKING FOR MICROCOMPUTER HARDWARE OR SOFTWARE? LOOK NO FURTHER THAN COMPUTAMART!

WARWICKSHIRE

YORKSHIRE

Please include my business details in the next available issue of Computing Today:

Business Name: ....

Address: ....

....

....

Tel. No.: ....

Open Hrs: ....

Contact (Office Use Only): ....

Post To: Computamart, Computing Today, 145 Charing Cross Rd., London WC2H 0EE.

ONLY £17.50!

AVAILABLE NOW

DRAGON 32 OWNERS

AVAILABLE NOW

Now you can buy a professional disk system to turn your Dragon into a real computer!

NOW DOUBLE-DENSITY DELTA DISK SYSTEM

THE DELTA DISK SYSTEM gives you . . .

- ★ An affordable disk system
- ★ Powerful DELTA disk commands — see right
- ★ Random, Sequential and Indexed file-handling
- ★ Simple plug-in to DRAGON — unlike other systems there are NO HARDWARE MODS needed to run DELTA!
- ★ Easily expandable 180K to 1.4 megabyte storage
- ★ Full range of business, utility and games software close to release
- ★ The price you see is the price you pay — NO HIDDEN RAM upgrade costs
- ★ Uses under 2K of user-RAM as DELTA is held in EPROM

FULL RANGE OF BUSINESS SOFTWARE AVAILABLE

SEND SAE FOR DETAILS

NEW TOOLKIT NOW AVAILABLE

DELTA CARTRIDGE — contains DELTA Disk Operating System, User Manual, demonstration diskette	£120.00
DELTA 1 — DELTA Cartridge, User Manual, a single-sided 40-track (180K) drive plus free cable	£320.00
DELTA 2 — as DELTA 1, but with a single-sided 80-track (360K) drive	£355.00
Disk Interface cable (supplied free with DELTA 1 or 2)	£9.95
ENCODER 09 assembler/disassembler/editor — integral with DELTA	£34.95
HOME ACCOUNTS full home package for DELTA	£14.95
INFORM — Data Base Management System commissioned especially for DELTA systems	£39.95

TOOLKIT for Dragon 32

Superb new toolkit cartridge now available which contains over 50 new commands: Full Screen Editor, Auto, Block, Break-Off, Break-On, Cappend, CDIR, CLSx, Deek, Doke, Echo, ENCA, ENCB, ERL, ERR, Error, Error Goto, Error Off, E.n, E, E;, Frame, Free$, Home, Hlin, Inkey, J.S.R., Jump, Key, T$, Limit, Look, Lvars, More, Old, Page, Pause, Prog, Prune, Rd, Replace, Resume, Resume Next, Resume Goto, Search, Set, Swap, Trace On, Trace Off, Vars, Vlin, Wild.

Available now —	Cartridge	£29.95
	DELTA enhancement	£29.95

ENCODER 09 — is a full symbolic assembler using standard mnemonics and pseudo op-codes. Source code can be incorporated into BASIC programs. The monitor section contains commands to allow memory display, modification and execution. Memory block move, breakpoint handling, full disassembly and a full editor are only a few of its many features. The most powerful assembler/disassembler/editor available for the DRAGON 32. Available as either an integral DELTA fitment or on cassette.
Cartridge £39.95. Disk — see above.

HIPRINT — screen dumper.

— will dump the entire contents of your DRAGON 32 high-res screen to a high resolution printer. Can be used for design, display etc (see left). Available at present for EPSON printers only. Other modules to follow shortly.
Tape £13.95. DELTA disk £14.95

POSTAGE and PACKING
Software 95p. DELTA systems £4.50.
ALL PRICES INCLUDE VAT.
Send SAE for full catalogue.

PREMIER MICROSYSTEMS

208 Croydon Road, Anerley, London SE20 7YX.

Access

TELEPHONE 01-659-7131 or 778-1706

Dealer and Overseas orders welcomed

BARCLAYCARD VISA

1978
1979
ATARI 800